**Anàlisis y clasificación de reseñas de películas**

**1. Introducción**

Hoy en día, la gente escribe constantemente en internet: comenta en redes sociales, participa en foros y publica reseñas sobre todo tipo de cosas. Entre estos textos, las reseñas de películas son especialmente interesantes, ya que suelen estar cargadas de opiniones, emociones y valoraciones personales. Analizar automáticamente este tipo de contenido abre la puerta a muchas aplicaciones dentro del Procesamiento del Lenguaje Natural (NLP), como detectar el tono de una reseña, predecir la puntuación que le han dado a la película o incluso recomendar otras que podrían gustarles.

En este trabajo, nos proponemos abordar varias de estas tareas empleando técnicas de NLP y aprendizaje automático. A partir de una colección de reseñas extraídas mediante web scraping de la web *metacritic.com*, construimos un conjunto de datos sobre el cual aplicamos diferentes fases del NLP: limpieza y preprocesamiento textual, análisis exploratorio, extracción de características, modelado y evaluación. Asimismo, exploramos tanto modelos entrenados desde cero como el uso de modelos preentrenados, con el objetivo de comparar su desempeño.

**2. Objetivos**

Los objetivos principales del trabajo son los siguientes:

1. **Extracción de datos**:
   * Obtener una base de datos de reseñas de películas mediante técnicas de web scraping sobre la web *metacritic.com*.
2. **Preprocesamiento de texto**:
   * Limpiar, normalizar y preparar el texto de las reseñas para su uso en modelos de NLP.
3. **Clasificación de sentimientos**:
   * Entrenar modelos de machine learning que clasifiquen automáticamente las reseñas en positivas, neutras o negativas.
   * Comparar el rendimiento de diferentes modelos, tanto entrenados desde cero como preentrenados.
4. **Predicción de puntuaciones**:
   * Estimar la puntuación numérica (de 0 a 10) de una reseña a partir de su contenido textual.
5. **Recomendación de películas**:
   * Desarrollar un sistema de recomendación que, a partir del historial de un usuario, sugiera nuevas películas que podrían interesarle.

**3. Métodos**

En esta sección se describen las distintas etapas que componen el desarrollo del proyecto, desde la recolección de los datos hasta la aplicación de modelos de aprendizaje automático.

**3.1. Web scrapping**

El proceso de recolección de datos comenzó definiendo un conjunto de páginas específicas del sitio web Metacritic que contienen listados de películas. Para cada página, se estableció un límite máximo de 24 películas a extraer. Para evitar bloqueos por parte del sitio web debido a accesos automáticos, se configuró una cabecera HTTP personalizada que simulaba ser un navegador real, utilizando un User-Agent común, como el de Google Chrome.

Para la extracción de los enlaces de las películas, se empleó la librería **BeautifulSoup** para procesar el HTML de las páginas y localizar los enlaces correspondientes. Solo se conservaron los enlaces de las primeras 24 películas por página, que luego se almacenaron en una lista maestra. A partir de esta lista, se generaron nuevas URLs dirigidas a las secciones de reseñas de cada película, incluyendo filtros específicos para focalizar el scraping en ciertos tipos de reseñas, como las mixtas ("Mixed Reviews") o las negativas ("Negative Reviews").

La extracción de las reseñas en sí se llevó a cabo utilizando **Selenium**, una herramienta que permite automatizar la interacción con navegadores reales. Esto fue necesario porque muchas de las reseñas no están presentes en el HTML estático, sino que se cargan dinámicamente mediante JavaScript. Selenium accede a cada URL de reseñas, espera a que el contenido se cargue completamente, y posteriormente analiza el HTML renderizado para extraer información clave de cada reseña: nombre del usuario, fecha de publicación, puntuación asignada y texto completo de la reseña.

El proceso de scraping se realizó en cuatro fases distintas. En la primera se recogieron reseñas positivas, neutras y negativas sin distinción, obteniendo un conjunto inicial de datos. En la segunda fase se amplió el número de muestras, ya que la cantidad inicial era insuficiente para un análisis robusto. Sin embargo, detectamos un desequilibrio importante en la distribución de clases, con muchas más reseñas positivas que neutras o negativas. Para corregir este sesgo, realizamos dos *scrappings* adicionales, filtrando exclusivamente reseñas neutras y negativas para equilibrar el conjunto de datos.

En total, se extrajeron aproximadamente 10,000 reseñas. Durante el proceso, se limitó la extracción a un máximo de 50 reseñas por película para evitar saturar el sistema. Todas las reseñas recopiladas se combinaron en una estructura tabular, como un DataFrame de Pandas, en la que cada fila representa una reseña con sus atributos asociados.

Finalmente, esta tabla fue revisada para garantizar la calidad y coherencia de los datos, dejando el conjunto listo para las etapas posteriores del proyecto, como el preprocesamiento, análisis exploratorio y modelado.

**3.2. Preprocesamiento**

**Carga y consolidación de datos**

El procedimiento inicia con la lectura de cuatro archivos CSV que contienen reseñas de usuarios. Estos archivos son importados como DataFrames individuales (df1, df2, df3 y df4), cada uno correspondiente a las cuatro fases de web scrapping mencionadas antes. Todos estos conjuntos de datos se concatenan en un único DataFrame unificado. Para verificar la limpieza inicial, se detectan filas duplicadas y se imprimen para revisión.

**Extracción y transformación de información de URLs**

Se extrae el nombre de la película desde el enlace usando expresiones regulares, transformando el formato URL en un nombre legible y capitalizado.

**Filtrado y detección de idioma**

Se eliminan las filas en las que la reseña está vacía o contiene la frase "SPOILER ALERT", ya que estas no aportan valor al análisis de sentimiento. También se aplica una función personalizada que detecta el idioma de cada reseña utilizando un detector de lenguas. Este paso considera excepciones, como textos vacíos o no identificables. Tras identificar los idiomas, se filtra el DataFrame para conservar únicamente aquellas reseñas escritas en inglés.

**Asignación de etiquetas de sentimiento**

Se introduce una nueva columna llamada *sentiment*, derivada de la puntuación (score) otorgada por el usuario. Se utiliza un criterio simple: reseñas con puntuaciones de 0 a 3 se clasifican como negativas, de 4 a 6 como neutrales y de 7 a 10 como positivas. Este el mismo criterio que sigue la página web, por lo que nos servirá para evaluar modelos más adelante.

**Análisis de "stopwords" y visualización**

Se importa un conjunto de palabras vacías en inglés (*stopwords*) desde la librería spaCy. Luego se define una función que calcula el porcentaje de palabras vacías dentro de cada reseña.

A continuación, se define otra función que genera un gráfico de barras que compara el porcentaje de palabras vacías y palabras con contenido real según la categoría de sentimiento. Esto permite observar, por ejemplo, si las reseñas negativas tienden a usar más palabras vacías que las positivas, o viceversa.

Gráfico, Gráfico de barras

El contenido generado por IA puede ser incorrecto.

Observamos que el porcentaje de *stopwords* en las reseñas es relativamente bajo, lo que indica que su eliminación no tendrá un impacto significativo en la longitud de los textos. Además, esta proporción se mantiene bastante equilibrada entre las tres clases de sentimiento.

**Normalización y lematización de texto**

Se establece una función para limpiar a fondo cada reseña. Este proceso incluye:

* Eliminación de números.
* Tokenización y lematización usando spaCy.
* Eliminación de puntuación, espacios innecesarios y stopwords.
* Conversión a minúsculas.
* Reconstrucción del texto como una única cadena con las palabras procesadas.

Esta función se aplica directamente sobre la columna *review*, reemplazando el texto original con su versión limpia y estructurada. Este paso es fundamental antes de aplicar modelos de análisis de texto, ya que estandariza el contenido y reduce ruido.

Cabe destacar que en la función de preprocesamiento se incluyó la opción de aplicar lematización como un parámetro configurable. Esto permitió generar, para cada reseña, tanto una versión lematizada del texto como otra sin lematizar. Esta decisión se tomó porque algunos de los modelos utilizados funcionan mejor cuando reciben el texto original, ya que incluyen sus propios mecanismos internos de lematización.

**3.3. Análisis exploratorio**

**Tokenización con spaCy**

Empezamos aplicando nlp (el modelo de spaCy) sobre cada reseña, creando una nueva columna llamada tokens que contiene los objetos Token generados por spaCy. Estos tokens son los elementos básicos para realizar análisis posteriores.

**Análisis de longitud de textos**

Se implementa una función para analizar la cantidad de palabras por reseña. A cada texto se le calcula su longitud en número de tokens. Luego, se generan histogramas para visualizar la distribución de longitudes por cada clase de sentimiento (NEG, NEU, POS). Se muestra también una distribución global que agrupa todas las clases juntas.

Gráfico, Histograma

El contenido generado por IA puede ser incorrecto.

Observamos que la longitud media de las reseñas ronda los 50 tokens.

**Nubes de palabras (WordClouds)**

Otra técnica de visualización textual utilizada es la nube de palabras. Para resaltar los términos más distintivos por clase. En esta ocasión preferimos elegir un enfoque distinto y personalizado para medir estas palabras más representativas siguiendo una serie de pasos.

* 1. Vectorización de texto con TF-IDF: Se transforma el texto en una representación que indica la importancia del texto en el documento, la métrica TF-IDF nos dará una matriz donde se mida cuántas veces aparece una palabra en el texto (TF – Term Frecuency) y penaliza las palabras que aparecen en muchas frases (IDF – Inverse Document Frecuency).
  2. Promedio de TF-IDF por categoría: Se calcula el valor TF-IDF para cada palabra dentro de cada categoría de la variable ‘sentiment’, lo que nos dará una idea de qué tan representativa es cada palabra dentro de una clase específica
  3. Cálculo de score diferencial: Este enfoque todavía daba bastantes palabras repetidas entre clases, por lo que para destacar las palabras más distintivas de cada clase y no solo las más frecuentes se crea un score para ello.

A partir de este score podemos averiguar las palabras más representativas y exclusivas para cada clase. Una vez calculado obtenemos los siguientes gráficos.

A close-up of words

AI-generated content may be incorrect.

**Generación y visualización de n-gramas (bigramas y trigramas)**

Se analizan secuencias frecuentes de palabras consecutivas:

**Bigramas**: Secuencias de 2 palabras (como “very good” o “not bad”).

**Trigramas**: Secuencias de 3 palabras (como “i really liked” o “not worth watching”).

El procedimiento incluye:

1. Extraer todos los tokens por clase de sentimiento.
2. Calcular la frecuencia de aparición de cada bigrama y trigramas.
3. Seleccionar los 10 más comunes por clase.
4. Visualizarlos en gráficos de barras horizontales, uno por clase, para facilitar la comparación.

**Gráfico

El contenido generado por IA puede ser incorrecto.**

Gráfico, Gráfico de barras

El contenido generado por IA puede ser incorrecto.

Observamos que los resultados obtenidos tienen bastante sentido. En las reseñas positivas suelen aparecer secuencias como “good movie” o “good movie year”, mientras que en las negativas frecuentan más secuencias como “bad movie” o “waste time money”, lo cual puede ser un indicativo de que hay buena coherencia en las reseñas y que están bien etiquetadas. Por otro lado, en las neutras se puede observar que hay una mezcla entre secuencias positivas y negativas, aunque claramente predominan las positivas.

**Distribución de muestras por clase**

Se han generado diagramas de barras tanto para las etiquetas de sentimiento (positivo, negativo y neutro) como para las puntuaciones (*score*), con el objetivo de analizar si las clases están balanceadas.

Gráfico, Gráfico de barras

El contenido generado por IA puede ser incorrecto. Gráfico, Histograma

El contenido generado por IA puede ser incorrecto.

Se observa un claro desbalance en el conjunto de datos: la mayoría de las reseñas son positivas, mientras que las neutras son las menos representadas. En cuanto a las puntuaciones (*scores*), destaca la frecuencia de la puntuación máxima (10), así como un pico notable en el valor mínimo (0). Esto sugiere una tendencia de los usuarios a emitir valoraciones extremas.

**3.4. Extracción de características**

Con el objetivo de transformar el texto en un formato numérico que pueda ser interpretado por los modelos de aprendizaje automático, se exploraron dos enfoques principales de extracción de características: representaciones dispersas (*sparse*) y representaciones densas (*dense*). Esto permitió comparar el rendimiento de los modelos según el tipo de representación utilizada.

**Características *sparse***

Para las representaciones dispersas, se empleó la técnica TF-IDF, que asigna un peso a cada palabra según su frecuencia en una reseña y su rareza en el conjunto completo de textos. Esto da lugar a vectores de alta dimensionalidad pero con la mayoría de valores en cero (baja densidad). Tras analizar el vocabulario total de las reseñas, se identificaron 22.891 palabras únicas. A partir de este dato, se estableció el parámetro max\_features=1000, de modo que el modelo solo considera las 1.000 palabras más frecuentes, reduciendo así la dimensionalidad y el ruido.

**Características densas**

En cuanto a las representaciones **densas**, se exploraron dos enfoques:

* **GloVe**: se empleó el modelo preentrenado de 50 dimensiones. Dado que GloVe proporciona embeddings a nivel de palabra, se calculó el promedio de los vectores de cada reseña para obtener una representación única por texto.
* **BERT**: se utilizó el modelo multilingüe *'all-mpnet-base-v2'* de *sentence-transformers*, el cual permite generar directamente un embedding para cada reseña completa, capturando de forma más precisa el contexto del texto. Esta alternativa se considera más adecuada que métodos como GloVe, donde los embeddings de palabras se promedian, lo que puede conllevar una pérdida de información contextual. Además, estos nuevos embeddings tienen una dimensión de 768, lo que representa una mayor capacidad de representación en comparación con los embeddings de 50 dimensiones que ofrece GloVe.

**3.5. Modelos**

Una vez extraídas las características de las reseñas, el siguiente paso consiste en aplicar distintos modelos de clasificación. En esta sección se describen los modelos utilizados, que incluyen tanto algoritmos tradicionales entrenados desde cero como modelos preentrenados.

**Modelos entrenados**

- **TF-IDF + Regresión Logística (LR):**  
Se utiliza la representación TF-IDF para convertir las reseñas en vectores numéricos dispersos, que luego se introducen en un modelo de regresión logística. Este modelo sirve como línea base por su simplicidad y eficacia en tareas de clasificación de texto.

- **TF-IDF + Topic Modeling (LSA) + LR:**  
A partir de los vectores TF-IDF, se aplica un modelo de Análisis Semántico Latente (LSA) para reducir la dimensionalidad de 1000 características a 15 capturando temas latentes en las reseñas. Los vectores resultantes se utilizan como entrada para un modelo de regresión logística.

- **TF-IDF + Topic Modeling (LSA) + XGBoost:**  
Similar al modelo anterior, pero en lugar de usar regresión logística, se emplea el algoritmo XGBoost, un potente modelo de boosting que puede capturar relaciones no lineales entre características.

- **GloVe Embeddings + LR:**  
Se representan las reseñas mediante el promedio de embeddings preentrenados GloVe de dimensión 50, generando vectores densos para cada reseña. Estos vectores se introducen en un modelo de regresión logística para realizar la clasificación.

- **BERT Embeddings + LR:**  
Se usan embeddings generados por el modelo preentrenado BERT (all-mpnet-base-v2), que proporciona directamente un vector denso para cada reseña. Estos vectores capturan el contexto completo del texto y se utilizan como entrada para un modelo de regresión logística.

**Modelos preentrenados**

- **Pipeline con modelo RoBERTa (*cardiffnlp/twitter-roberta-base-sentiment*):**  
Se utilizó una pipeline de Hugging Face que incorpora un modelo RoBERTa preentrenado específicamente para análisis de sentimiento en textos de Twitter. Este modelo permite obtener predicciones directamente, sin necesidad de entrenamiento adicional.

- **BERT preentrenado con fine-tuning (*google-bert/bert-base-cased*):**  
Se parte de un modelo BERT base preentrenado y se ajusta (fine-tuning) a nuestro conjunto de datos mediante entrenamiento adicional. Se adapta el número de etiquetas al problema específico (positivo, negativo, neutro) para que el modelo aprenda a clasificar según nuestras categorías.

**4. Resultados**

En esta sección se presentan los resultados obtenidos por los distintos modelos descritos en la sección anterior. Primero se analizará el rendimiento de los clasificadores entrenados para predecir el **sentimiento** de las reseñas (positivo, negativo o neutro), y posteriormente se evaluarán los modelos desarrollados para predecir la **puntuación numérica** (score de 0 a 10) asignada por los usuarios.

**Modelos de clasificación de sentimiento**

**Modelos entrenados**

**1. TF-IDF + Regresión Logística (LR)**

* *Accuracy:* ~69%
* Buen rendimiento para clases NEG y POS (f1-score 0.71 y 0.79 respectivamente).
* Clase NEU más difícil, f1-score 0.43, con menor recall (0.35).

**2. TF-IDF + LSA + LR**

* *Accuracy:* ~61%
* Reducción de performance respecto al modelo anterior.
* NEU muy baja recall (0.12) y f1 muy baja (0.18).
* Parece que la reducción dimensional con LSA afectó negativamente la clasificación de la clase neutral.

**3. TF-IDF + LSA + XGBoost**

* *Accuracy:* ~66%
* Mejora respecto al modelo anterior con LSA + LR.
* Recupera recall mejor en POS (0.83) y mejor f1 en NEG y POS.
* Clase NEU sigue siendo la más débil (f1 0.42).
* XGBoost captura relaciones no lineales y mejora resultado.

**4. GloVe Embeddings + LR**

* *Accuracy:* ~63%
* Rendimiento moderado, con f1-score 0.66 en NEG y 0.74 en POS.
* NEU tiene recall bajo (0.19) y f1 0.27, sigue siendo clase difícil.

**5. BERT Embeddings + LR**

* *Accuracy:* ~68%
* F1 similar a TF-IDF + LR, buena recuperación en POS (0.78) y NEG (0.72).
* NEU con mejor recall (0.31) y f1 0.40, mejora sobre modelos previos.
* Embeddings contextuales aportan mejor representación semántica.

**6. BERT Embeddings + LR (class\_weight=balanced)**

* *Accuracy:* ~69%
* Equilibrio entre precisión y recall en NEG y POS.
* NEU mejora notablemente (f1 0.47) respecto a modelos anteriores.

**Modelos preentrenados**

**7. Pipeline RoBERTa preentrenado (cardiffnlp/twitter-roberta-base-sentiment)**

* *Accuracy:* ~62%
* Rendimiento general más bajo.
* NEU con f1 alrededor de 0.34, POS y NEG moderados.

**8. BERT preentrenado con fine-tuning (google-bert/bert-base-cased)**

* Accuracy: ~70%
* Rendimiento general sólido y balanceado.
* Mejor manejo de la clase NEU con f1 de 0.52 y recall de 0.58.
* Alta precisión en POS (f1 0.81) y NEG (f1 0.71).

**Conclusión general sobre los modelos de predicción de sentimiento**

Los modelos evaluados muestran un desempeño variable en la clasificación de sentimientos, con diferencias notables entre las técnicas tradicionales y las basadas en embeddings contextuales o modelos preentrenados.

* Entre los enfoques clásicos basados en representaciones dispersas, la combinación de **TF-IDF y regresión logística** ha ofrecido el mejor rendimiento, con *accuracies* cercanas al 69**%**. No obstante, la clase neutral ha resultado especialmente difícil de predecir con precisión. Por otro lado, la reducción dimensional mediante LSA tiende a deteriorar el rendimiento general, afectando en mayor medida a la detección de esta clase. En contraste, el uso de **XGBoost** mejora la capacidad del modelo para capturar relaciones no lineales, logrando mejores resultados en las clases positiva y negativa, aunque sin llegar a superar el rendimiento del modelo con TF-IDF y regresión logística.
* Entre los modelos basados en representaciones densas, **el embedding de BERT combinado con regresión logística** destaca por su rendimiento. Este muestra un rendimiento ligeramente superior al anterior (TF-IDF + regresión logística), especialmente al considerar métricas por clase. Ambos modelos tienen la misma **accuracy** (0.69), pero el segundo presenta mejoras clave:
  + - En la clase **NEG**, el *F1-score* pasa de **0.71** a **0.72**.
    - En la clase **NEU**, que suele ser la más difícil de clasificar, el *F1-score* mejora de **0.43** a **0.47**, y el *recall* sube de **0.35** a **0.42**.
    - En la clase **POS**, el rendimiento se mantiene estable con un *F1-score* de **0.79** en ambos casos.
* Finalmente, de los modelos preentrenados el mejor ha sido el **BERT preentrenado con fine-tuning,** que alcanza el mejor rendimiento global, con un accuracy del 70%. Ofrece mejores resultados especialmente en la clase **NEU**, sin sacrificar precisión en **NEG** o **POS**. Además, su mayor **macro** y **weighted F1-score** (ambos **0.68** y **0.71**, respectivamente) indican una mayor capacidad para generalizar entre clases, lo cual es crucial en tareas de clasificación multiclase como análisis de sentimiento.

**Modelos de predicción de puntuación**

Dado que la predicción de la puntuación de una reseña representa una tarea más compleja —al implicar 11 clases distintas (de 0 a 10) en lugar de solo tres como en la clasificación de sentimientos—, se optó por reutilizar los modelos entrenados que obtuvieron mejores resultados en la tarea de clasificación de sentimientos. Aunque se valoró la posibilidad de emplear un modelo preentrenado, no se encontró ninguno adaptado específicamente a la clasificación de textos en puntuaciones del 0 al 10.

**1. TF-IDF + Regresión Logística (LR)**

* *Accuracy:* ~30%

**2. BERT + Regresión Logística (LR)**

* *Accuracy:* ~30%

Calendario

El contenido generado por IA puede ser incorrecto.Calendario

El contenido generado por IA puede ser incorrecto.

Se observa que el rendimiento de ambos modelos en la tarea de predicción de puntuación es considerablemente bajo. En ambos casos, los modelos tienden a concentrar sus predicciones en los valores extremos (0 y 10), con un pico adicional en la puntuación intermedia de 6. Este comportamiento sugiere que el modelo podría estar asociando las reseñas neutras con dicha puntuación central. Al comparar estos resultados con la distribución de frecuencias observada en el análisis exploratorio, se confirma que las puntuaciones con mayor número de ejemplos son, precisamente, 0, 6 y 10. Esto indica que el modelo podría estar sesgado hacia las clases mayoritarias, lo que es habitual en contextos con desequilibrio de clases, y explica en parte su incapacidad para generalizar correctamente a las clases intermedias menos representadas.

**5. Mejoras**

**5.1. Análisis de errores de clasificación**

Para intentar mejorar el rendimiento de los modelos, empezamos esta fase revisando los errores de clasificación más frecuentes. En concreto, analizamos la matriz de confusión del modelo basado en TF-IDF y regresión logística, ya que fue uno de los que mejores resultados obtuvo en términos generales.

Gráfico, Gráfico de rectángulos

El contenido generado por IA puede ser incorrecto.

Este análisis nos confirmó algo que ya habíamos notado al revisar las métricas por clase: la clase **neutral (NEU)** es, con diferencia, la que más problemas da. La mayoría de las reseñas neutras se están clasificando como positivas, y en menor medida, como negativas. Esto puede tener cierta explicación si recordamos el análisis exploratorio inicial, donde vimos que muchos de los bigramas y trigramas más comunes en las reseñas neutras tenían un tono claramente positivo. Esto puede estar confundiendo al modelo, haciéndole pensar que ese tipo de texto corresponde a una reseña positiva.

Para entender mejor qué está pasando, decidimos revisar manualmente algunas reseñas que estaban etiquetadas como NEU, pero que el modelo clasificó como POS o NEG. A continuación, se muestran algunos ejemplos que nos parecieron especialmente representativos:

"*disappointed movie dialogue stupid rhythm movie slow people say movie perfect fail attempt homorromantic movie lay clichés high class lifestyle bore death apricot scene awkward disgusting soundtrack interesting"*

True label: NEU | Predicted label: NEG

En este caso la reseña está mal etiquetada. No es un fallo del modelo, sino del proceso de etiquetado basado únicamente en la puntuación numérica. El contenido textual es claramente negativo, y el modelo lo ha clasificado correctamente.

*"spotlight attractive plot it´s grip portray controversial story people film classic it´ film doesn´t generate sensation fear danger threat moment remember memorable dialogue character development think film recieve praise it´s simplicity storytelling controversial aspect tom mccarthy brave didn´t direct memorable enjoyable film it´s dry joyless"*

True label: NEU | Predicted label: POS

En este caso, tenemos una reseña que mezcla matices tanto positivos como negativos, por lo que pensamos que está correctamente etiquetada como neutra. El modelo, sin embargo, parece dar más peso a los aspectos positivos e ignora comentarios negativos importantes al final del texto, como *"dry"* y *"joyless"*. Por tanto, se trata de un error del modelo, que no logra captar adecuadamente el equilibrio tonal de la reseña.

*“give movie high score tarantino director high score”*

True label: NEU | Predicted label: NEG

En este caso, el problema parece doble. Por un lado, la etiqueta **neutral** no resulta del todo coherente, ya que la reseña menciona explícitamente *"give movie high score"*, lo que sugiere una valoración positiva. Por otro lado, la predicción del modelo como **negativa** tampoco es adecuada. Lo más probable es que el origen del error esté en lo poco informativa y ambigua que resulta la reseña: es muy breve, sin contexto ni contenido claro, lo que dificulta tanto una etiquetación precisa como una predicción fiable por parte del modelo.

En conclusión, las reseñas neutras —las que presentan peores resultados de clasificación— suponen un reto por diversos motivos. En algunos casos, el problema parece estar en el etiquetado, ya que ciertas reseñas catalogadas como neutras muestran un tono claramente positivo o negativo. En otros, el modelo no logra captar adecuadamente los matices propios de este tipo de opiniones. A continuación, se exploran algunas posibles soluciones con el objetivo de mejorar la clasificación de esta clase.

**5.2. Eliminación de neutras**

En primer lugar, se optó por eliminar las reseñas etiquetadas como neutras. Si bien somos conscientes de que se trata de una solución drástica y de utilidad limitada en la práctica, el objetivo era comprobar si prescindir de esta clase tan conflictiva suponía una mejora significativa en el rendimiento del modelo.

**1. TF-IDF + Regresión Logística (LR)**

* *Accuracy:* ~87%

**2. BERT + Regresión Logística (LR)**

* *Accuracy:* ~87%

Al eliminar la clase neutra, el rendimiento de ambos modelos mejora notablemente, con métricas más equilibradas y altas en precisión, recall y F1. Esto se debe a que las opiniones neutras suelen ser ambiguas y difíciles de clasificar, mientras que las polarizadas (positivas o negativas) ofrecen señales más claras para los modelos.

**5.3. Eliminación de reseñas poco coherentes**

Con el objetivo de mejorar la clasificación sin eliminar por completo la clase neutra, se planteó una estrategia alternativa basada en la detección y eliminación de reseñas poco coherentes. La hipótesis es que algunas reseñas contienen mensajes contradictorios o ambiguos con la etiqueta que les ha sido asignada.

Para identificar estas reseñas, se aplicó un análisis de polaridad frase a frase utilizando la librería **TextBlob**, que asigna una puntuación entre -1 (muy negativo) y 1 (muy positivo) a cada oración. Previamente, se segmentaron las reseñas en frases mediante la función sent\_tokenize de la librería nltk. Por cada oración, se calculó su puntuación de polaridad y se almacenó junto con su índice original y el nombre de la película.

Una vez extraídas todas las frases y sus polaridades, se calculó el **promedio del score** para cada reseña completa, agregando este valor como una nueva columna (mean\_score) al DataFrame original. Este promedio ofrece una medida aproximada de la orientación general del sentimiento de cada reseña.

Después, se visualizó la distribución de estos valores de polaridad media diferenciando por clase (POS, NEU, NEG), mediante un histograma:

Gráfico, Histograma

El contenido generado por IA puede ser incorrecto.

Establecimos umbrales específicos por clase para conservar únicamente aquellas reseñas cuya polaridad media fuera coherente con su etiqueta:

* Para las reseñas **positivas (POS)**, se exigió una polaridad media de al menos **0.1**.
* Para las reseñas **negativas (NEG)**, la polaridad media debía ser **igual o inferior a -0.1**.
* Para las reseñas **neutras (NEU)**, se aceptaron aquellas con polaridad media dentro del rango **(-0.3, 0.3)**, excluyendo las que tuvieran un sesgo claro hacia lo positivo o lo negativo.

Tras aplicar este filtrado, el número total de muestras se redujo de **8703 a 5990**, eliminando así reseñas cuya puntuación de polaridad no coincidía con la clase asignada, lo que podría introducir ruido en el entrenamiento del modelo.

El balance de clases ahora queda así:

* + - 3092 reseñas positivas
    - 1999 reseñas neutras
    - 899 reseñas negativas

Se observa que la mayor reducción de reseñas se ha producido en la categoría negativa, mientras que las categorías positiva y neutra han conservado una cantidad considerable de ejemplos.

Una vez filtrado el dataset probamos a clasificar de nuevo con algunos de nuestros modelos más exitosos.

**1. TF-IDF + Regresión Logística (LR)**

Calendario

El contenido generado por IA puede ser incorrecto.

**2. BERT + Regresión Logística (LR)**

Calendario

El contenido generado por IA puede ser incorrecto.

**4.BERT preentrenado con fine-tuning (google-bert/bert-base-cased)**

Una captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Observamos que la mejora es notable. Hemos mejorado el accuracy manteniendo buenas métricas en general para todas las clases. Podemos concluir que las reseñas eliminadas sí que estaban confundiendo al modelo. Es interesante que a pesar de habernos quedado con menos reseñas, la clasificación es bastante mejor. Podemos destacar como mejor modelo el obtenido mediante fine tuning tras eliminar las reseñas poco coherentes, que alcanza un 0.86 de accuracy.

**6. Recomendador**

Para complementar el análisis de sentimientos, desarrollamos un sistema de recomendación de películas personalizado. El objetivo era recomendar nuevas películas a un usuario concreto en función de los géneros que más le han gustado previamente, inferidos a partir de las puntuaciones (score) que ha otorgado en sus reseñas.

**1) Enriquecimiento del dataset**

Dado que el dataset original de reseñas no incluía información sobre los géneros de las películas, se integró un segundo dataset externo descargado de Kaggle. Este dataset contiene información detallada de aproximadamente 16.000 películas, incluyendo sus géneros. Se transformó esta información en formato "one-hot encoding", generando una matriz binaria donde cada columna representa un género y cada fila una película, con 1 si la película pertenece al género y 0 en caso contrario.

**2) Selección de usuario**

Se seleccionó aleatoriamente un usuario que hubiese escrito 10 reseñas. Esto garantiza una base mínima para poder construir su perfil de preferencias.

**3) Construcción del perfil del usuario**

Se filtraron las películas que este usuario ha reseñado y que además están presentes en el dataset con géneros. Eso redujo el número de películas de 10 a 6.

Estas son las películas reseñadas por el usuario que se considerarán para realizar la recomendación y su puntuación:

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

A partir de las puntuaciones asignadas por el usuario y los géneros asociados a esas películas, se construyó un vector de preferencias. Esto se logró calculando un producto escalar entre la matriz binaria de géneros y las puntuaciones dadas por el usuario, obteniendo así una estimación del "peso" o afinidad del usuario por cada género.

**4) Cálculo de recomendaciones**

A continuación, se calcularon puntuaciones para todas las películas del dataset en función de la afinidad del usuario por los distintos géneros. Cuanto más coinciden los géneros de una película con los géneros preferidos por el usuario, mayor será su puntuación. Estas puntuaciones se normalizaron dividiendo entre la suma total de pesos de género del usuario.

**5) Resultados**

Finalmente, se seleccionaron las cinco películas con mayor puntuación como las más recomendadas para ese usuario. El resultado es una tabla que muestra el título de cada película, su puntuación de recomendación y los géneros asociados.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Este sistema, aunque sencillo, permite adaptar las recomendaciones al perfil específico de cada usuario, basándose únicamente en sus gustos implícitos (géneros de películas valoradas positivamente), sin requerir feedback explícito adicional.