UNIVERSIDAD NACIONAL DE ROSARIO

FACULTAD DE CIENCIAS EXACTAS, INGENIERÍA Y AGRIMENSURA

TEC. INTELIGENCIA ARTIFICIAL

IA3.5## - Procesamiento del Lenguaje Natural

Trabajo práctico N°1

INTEGRANTES:

Alsop Agustín (A-4651/7)
Hachen Rocío (H-1184/3)

2024

1. Índice

2. Introducción

2.1. Contexto

Una persona dentro de un mes, se tomará 15 días de vacaciones en la playa. Sin embargo, se estima que durante al menos cuatro de esos días habrá lluvias, lo que podría limitar las actividades al aire libre. Para esos días de mal clima, se propone una solución que facilite la recreación en función del estado de ánimo del día.

2.2. Objetivo

Desarrollar un programa de Procesamiento de Lenguaje Natural que, según el estado de ánimo del usuario, recomiende entre ver una película, jugar un juego de mesa o leer un libro (o varias opciones para cada caso). Para ello, deberá construir un clasificador que categorice el estado de ánimo del usuario. Luego sugerir el conjunto de recomendaciones basada en una frase de preferencia ingresada por el usuario.

2.3. Breve descripción de la estructura del informe

Este informe se organiza en diversas secciones que abarcan desde la introducción hasta las conclusiones, con el fin de ofrecer una comprensión completa del proyecto desarrollado. A continuación, se detalla la estructura del documento:

Sección 2: Introducción

- En esta sección se proporciona el contexto general del proyecto, se establece el objetivo principal y se ofrece una breve descripción de la estructura del informe.

Sección 3: Metodología

- Aquí se abordan los datos utilizados, así como los métodos y técnicas aplicadas durante el desarrollo del proyecto. También se mencionan las herramientas y tecnologías que se emplearon para la implementación.

Sección 4: Desarrollo e Implementación

- Esta sección es la más extensa y está dividida en subapartados. Comienza con una introducción general y un resumen, seguido de un análisis detallado de las librerías utilizadas y las funciones generadas. Cada función se describe de manera individual, explicando su propósito y funcionamiento. Además, se incluye un apartado dedicado a la carga de

bases de datos, donde se detalla las técnicas utilizadas para cargar las base de datos a la aplicación..

Sección 5: Resultados

- En esta parte se presentan los resultados obtenidos a partir del análisis de datos y las recomendaciones generadas. Se incluye un análisis del modelo de regresión logística utilizado para la detección de emociones y se discuten los resultados de la recomendación de actividades. También se evalúa el rendimiento del código implementado.

Sección 6: Conclusiones

- Esta sección ofrece un resumen de los resultados generales, verifica si se han alcanzado los objetivos propuestos y brinda recomendaciones para futuras investigaciones o mejoras en el sistema.

Sección 7: Referencias

- Aquí se citan todas las fuentes y materiales consultados durante el desarrollo del proyecto.

Sección 8: Anexos

- Finalmente, se presentan anexos que pueden contener información adicional, gráficos, tablas u otros datos relevantes que complementan el contenido del informe.

Esta estructura está diseñada para facilitar la comprensión del proyecto y guiar al lector a través de los diferentes aspectos del mismo, asegurando que se aborden de manera coherente y ordenada.

3. Metodología

3.1. Datos Utilizados

Para la realización de este trabajo, se utilizaron cuatro bases de datos, las cuales se describen a continuación:

- **bgg_database.csv:** Esta base de datos contiene información detallada sobre juegos de mesa, incluyendo títulos, categorías, puntuaciones y otros atributos relevantes. Es útil para el análisis de tendencias y popularidad en el ámbito de los juegos de mesa.

- **IMDB-Movie-Data.csv:** Una base de datos centrada en películas, que abarca detalles como títulos, géneros, años de lanzamiento, puntuaciones de IMDb, y más. Permite realizar análisis de preferencias y tendencias dentro de la industria cinematográfica.

- **Libros del Proyecto Gutenberg:** Esta base de datos fue creada mediante web scraping para recopilar información sobre los 1000 libros más populares del Proyecto Gutenberg. Contiene datos como títulos, autores y el número de descargas, facilitando el estudio de obras literarias de dominio público. El enlace utilizado para obtener esta información es el siguiente: Proyecto Gutenberg - Libros Populares.

- **db_emotion:** Base de datos que contiene frases clasificadas en tres categorías de emoción: triste, neutro y feliz. Esta información es valiosa para el análisis de emociones y puede ser utilizada en proyectos de procesamiento de lenguaje natural y estudios relacionados con la detección de emociones en textos.

3.2. Métodos y técnicas

Para el cumplimiento del objetivo determinamos 3 áreas en las que se utilizarán técnicas de NLP:

1. Para detectar las emociones
2. Para detectar si el usuario se encuentra en situación de riesgo emocional
3. Para recomendar una película, libro o juego de mesa

3.2.1. Detección de emociones

Para poder identificar la emoción del usuario en el primer prompt que este ingresa, se utilizó un modelo de regresión logística. Este modelo de clasificación se entrena para predecir emociones a partir de representaciones vectoriales de las

frases, generadas por el modelo preentrenado **all-mpnet-base-v2** de la biblioteca **SentenceTransformer (SBERT)**.

Primero, los datos fueron preprocesados convirtiendo las frases a minúsculas. A continuación, el dataset se dividió en conjuntos de entrenamiento y prueba, asignando el 90% de los datos para entrenamiento y el 10% para evaluación.

Para obtener una representación semántica de cada frase, se generaron embeddings, usando **model.encode**, que convierte las frases de texto en vectores numéricos que capturan el significado y contexto de las palabras. Estos embeddings fueron usados como entradas para el modelo de **Regresión Logística**.

La regresión logística se entrenó usando el solver **lbfgs**, configurado para admitir múltiples clases, lo cual permite que el modelo prediga entre varias emociones posibles (mapeadas como -1 (triste), 0 (neutro) y 1 (feliz).

3.2.2. Detección de riesgo

Para detectar si el usuario se encuentra en situación de riesgo emocional, se implementó un mecanismo de detección de frases peligrosas que busca identificar indicios de pensamientos autodestructivos o de desesperanza en las respuestas del usuario.

Se utilizó el modelo de procesamiento de lenguaje, **spaCy** en español, **es_core_news_lg**, que permite analizar y comparar el contenido semántico de las frases mediante el cálculo de similitudes entre frases.

En primer lugar, se definió un conjunto de frases de alto riesgo que representan expresiones comunes de pensamientos suicidas o autolesivos, tales como "quiero morir". La función **redFlagIdentifier** recibe como parámetro una frase y evalúa su similitud con cada una de las frases peligrosas definidas.

Para cada frase ingresada, el código procesa la frase usando **spaCy** y calcula la similitud entre la frase ingresada y cada frase peligrosa mediante el método **similarity**, que evalúa la proximidad semántica en el espacio vectorial.

Si alguna frase peligrosa tiene una similitud superior al umbral definido (0.7 en este caso), la función retorna **True**, indicando una posible situación de riesgo emocional.

Se puede leer más sobre su funcionamiento en la sección 4.4.1.

3.2.3. Recomendación de actividad

Se le pide al usuario que ingrese un **prompt** donde describa el tipo de actividad que le gustaría ver, como una película, libro o juego de mesa. A partir de ese prompt, el sistema utiliza el modelo **distiluse-base-multilingual-cased-v1** de **SentenceTransformer (SBERT)**, el cual genera *embeddings* comparables entre múltiples idiomas, incluyendo inglés y español.

Este *embedding* se compara con los *embeddings* de la base de datos de actividades mediante la función de similitud coseno, lo que permite identificar las actividades que más se asemejan al prompt ingresado.

Luego, el sistema ordenará las coincidencias de mayor a menor similitud y devuelve una lista de recomendaciones.

3.3. Herramientas y tecnologías

Para la realización del trabajo, se utilizaron las siguientes herramientas:

- **Python:** Python es un lenguaje de programación de alto nivel que destaca por su sintaxis clara y su facilidad de uso, lo que permite un desarrollo rápido y eficiente. Se utilizó principalmente para la implementación de scripts y modelos, aprovechando su amplia biblioteca de paquetes para manipulación de datos y análisis, así como su versatilidad en entornos de programación científica.

- **Google Colab:** Google Colab es una plataforma en la nube que permite programar y ejecutar código en Python de manera interactiva. Ofrece una infraestructura accesible para realizar cálculos complejos sin depender de recursos locales y facilita la colaboración en tiempo real. Esta herramienta fue clave para el desarrollo y prueba de los scripts, permitiendo ejecutar el código y visualizar los resultados sin necesidad de una instalación local.

- **Google Docs:** Google Docs es una herramienta de procesamiento de textos en línea que facilita la creación y edición de documentos. En este proyecto, se utilizó para redactar y estructurar el informe, permitiendo el trabajo colaborativo y la edición en tiempo real, además de ofrecer opciones de formato y revisión de contenido.

- **ChatGPT**: ChatGPT es una herramienta de inteligencia artificial desarrollada por OpenAI, que utiliza procesamiento de lenguaje natural para generar y entender texto. En este proyecto, se empleó para la creación de la base de datos **db_emotion**, proporcionando frases clasificadas en tres emociones: triste, neutro y feliz. ChatGPT facilitó la generación de datos

simulados de manera rápida y precisa, contribuyendo así a la preparación del dataset necesario para el análisis emocional.

4. Desarrollo e Implementación

4.1. Introducción

Para alcanzar los objetivos planteados en este informe, se desarrolló **WILSON**, una aplicación diseñada para el reconocimiento de emociones y la recomendación de actividades lúdicas. **WILSON** actúa como un acompañante digital, proporcionando sugerencias personalizadas tanto en momentos felices como en aquellos más tristes, adaptándose a las emociones del usuario para mejorar su experiencia y bienestar.

4.2. Resumen

Al iniciar WILSON, el sistema le solicita al usuario que exprese cómo se siente. Utilizando un modelo de regresión logística entrenado con el dataset db_emotion, WILSON detecta la emoción del usuario (como feliz, neutro o triste) y, en función de ella, adapta su respuesta. A continuación, el usuario debe ingresar una frase que describa sus preferencias sobre lo que le gustaría ver, jugar o leer en ese momento. Con esta información, WILSON recomienda una película, libro o juego que mejor se ajuste a sus necesidades emocionales, proporcionando así una experiencia personalizada y adecuada a su estado de ánimo.

4.3. Librerías utilizadas

Sklearn: Scikit-learn (sklearn) es una biblioteca de Python que proporciona herramientas eficientes y accesibles para el aprendizaje automático. Incluye una variedad de algoritmos para clasificación, regresión y agrupación, así como herramientas para preprocesamiento de datos y evaluación de modelos.

- **Regresión Logística:** Es un modelo de clasificación dentro de sklearn que permite predecir la probabilidad de una categoría binaria o multiclase. En este proyecto, se utiliza para clasificar las emociones en diferentes categorías.
- **train_test_split:** Función de sklearn que divide los datos en conjuntos de entrenamiento y prueba, permitiendo evaluar la precisión del modelo en datos que no ha visto antes.
- **metrics:** Módulo de sklearn que incluye herramientas para evaluar el rendimiento de los modelos, como precisión, sensibilidad y otras métricas de clasificación.
- **accuracy_score:** Función en sklearn que calcula la precisión del modelo, midiendo el porcentaje de predicciones correctas en el conjunto de prueba.

- **classification_report:** Genera un reporte detallado de las métricas de clasificación, incluyendo precisión, recuperación y puntaje F1 para cada clase, ayudando a analizar el rendimiento del modelo.

Sentence Transformer: Biblioteca que convierte frases en representaciones vectoriales mediante modelos de aprendizaje profundo. Se utiliza para medir similitudes entre frases, útil en el reconocimiento de emociones y en la recomendación de contenido basado en frases de entrada del usuario.

numpy: Biblioteca fundamental en Python para el manejo de arrays y operaciones matemáticas. Ofrece herramientas de cálculo eficientes y es utilizada para manipular datos numéricos en grandes cantidades.

pandas: Biblioteca que proporciona estructuras de datos y herramientas de análisis de datos flexibles y fáciles de usar. Facilita la carga, manipulación y análisis de bases de datos como tablas y archivos CSV.

thefuzz (procesamiento de coincidencias de texto): Permite comparar cadenas de texto y calcular su similitud mediante el procesamiento difuso. En este proyecto, es útil para hacer coincidir frases o palabras ingresadas por el usuario con palabras clave en la base de datos de recomendaciones.

bs4 (BeautifulSoup): Biblioteca para el análisis y extracción de datos de documentos HTML y XML. Se utilizó para realizar web scraping y extraer información de sitios web, como el Proyecto Gutenberg, para crear la base de datos de libros.

spacy: Biblioteca de procesamiento de lenguaje natural que ofrece herramientas avanzadas para el análisis de texto. Permite la tokenización, reconocimiento de entidades y etiquetado de partes de oración, ayudando a procesar y entender las frases de los usuarios en el proyecto.

torch (PyTorch): Biblioteca de aprendizaje profundo que facilita la creación y entrenamiento de modelos de redes neuronales. PyTorch se usa junto con transformers para implementar modelos de procesamiento de lenguaje natural.

nltk: Biblioteca de procesamiento de lenguaje natural que ofrece herramientas para el análisis de texto, tokenización, etiquetado y más. Ayuda a preprocesar y analizar las frases de los usuarios en la detección de emociones.

transformers: Biblioteca de Hugging Face que permite cargar y utilizar modelos de lenguaje avanzados, como BERT. Facilita el uso de modelos preentrenados en tareas de NLP como clasificación, generación de texto y análisis de emociones.

- **BertTokenizer:** Tokenizador específico de BERT que convierte frases en tokens compatibles con el modelo BERT, preparándolas para su procesamiento.
- **BertModel:** Implementación del modelo BERT (Bidirectional Encoder Representations from Transformers), que genera representaciones contextuales de texto. Se utiliza en el proyecto para comprender mejor el contenido de las frases ingresadas por el usuario.

4.4. Funciones generadas

4.4.1. redFlagIdentifier

Descripción: Esta función verifica si una frase ingresada contiene indicios de pensamientos o expresiones relacionadas con autolesiones o suicidio, utilizando procesamiento de lenguaje natural (NLP) con la biblioteca spaCy.

Parámetros:

- frase: La frase ingresada por el usuario que se analizará en busca de patrones peligrosos.

Funcionamiento:

1. La función carga el modelo de lenguaje en español (es_core_news_lg) de spaCy para procesar la frase ingresada y las frases de referencia (contenidas en frases_peligrosas).
2. Convierte la frase del usuario a minúsculas para asegurar la consistencia y genera un objeto doc con spaCy.
3. Compara la similitud semántica de la frase del usuario con cada frase en la lista frases_peligrosas usando el método .similarity().
4. Si la similitud con alguna frase supera el umbral de 0.7, la función devuelve True, indicando que se ha detectado una expresión potencialmente peligrosa.
5. Si no encuentra ninguna coincidencia, devuelve False.

Propósito:

Esta función es útil para identificar frases que podrían requerir atención o intervención, proporcionando una medida de seguridad y apoyo emocional en la aplicación.

4.4.2. analisisEmocion

Descripción:

Esta función analiza el estado emocional del usuario con base en el texto ingresado, utilizando un modelo de regresión logística (modeloLR) para clasificación de emociones y el modelo Sentence Transformer (modeloST) para transformar la frase en una representación vectorial.

Parámetros:

- input_usuario: La frase ingresada por el usuario que se evaluará para detectar su emoción.
- modeloLR: El modelo de regresión logística preentrenado que clasifica la emoción de la frase.
- modeloST: El modelo Sentence Transformer utilizado para convertir el texto en un vector numérico para el análisis.

Funcionamiento:

1. Convierte la frase del usuario a minúsculas para asegurar la consistencia en el preprocesamiento.
2. Usa modeloST para vectorizar la frase, transformándola en una representación numérica adecuada para el modelo de clasificación.
3. Clasifica la emoción de la frase mediante modeloLR, prediciendo entre tres categorías: triste (-1), neutro (0) y feliz (1).
4. Muestra el texto ingresado y la emoción predicha en la consola.
5. Retorna la clasificación de la emoción (triste, neutro, o feliz) como un valor numérico.

Propósito:

analisisEmocion permite identificar el estado emocional del usuario en tiempo real, proporcionando un resultado categórico que se puede utilizar para personalizar recomendaciones y respuestas en función del estado de ánimo detectado.

4.4.3. yesNoValidator

Descripción:

Esta función permite validar respuestas de sí o no ingresadas por el usuario. Acepta varias variantes de cada respuesta (como "Y", "Yes", "Si", etc.), verificando que la entrada sea válida antes de continuar.

Funcionamiento:

1. Solicita al usuario que ingrese una respuesta (sí o no) con el mensaje "Responda si o no (Y/N):".
2. Define dos listas de respuestas válidas:
3. yes_answers: Contiene las variantes para respuestas afirmativas, como "Y", "y", "Yes", "Si", "S", "s" y ":)".
4. no_answers: Contiene las variantes para respuestas negativas, como "N", "n", "No" y ":(".
5. Mientras la respuesta del usuario no esté en valid_answers (que contiene tanto respuestas afirmativas como negativas), solicita una respuesta válida mostrando todas las opciones permitidas.
6. Si la respuesta está en yes_answers, devuelve True; si está en no_answers, devuelve False.

Propósito:

yesNoValidator garantiza que las respuestas sean consistentes y ayuda a evitar errores o ambigüedades en el input del usuario, mejorando la interacción y el flujo de la aplicación.

4.4.4. saludo

Descripción:
Esta función genera un saludo contextualizado según la hora local de Buenos Aires, Argentina, devolviendo un mensaje de buenos días, buenas tardes o buenas noches, junto con la hora exacta en formato de 24 horas.

Funcionamiento:

1. Establece la zona horaria local a "America/Argentina/Buenos_Aires" usando la biblioteca pytz.
2. Obtiene la hora actual en la zona horaria de Buenos Aires.
3. Según el valor de la hora:
4. Entre las 6:00 y las 11:59, el saludo es "¡Buenos días!".
5. Entre las 12:00 y las 19:59, el saludo es "¡Buenas tardes!".
6. Desde las 20:00 hasta las 5:59, el saludo es "¡Buenas noches!".
7. Convierte la hora actual al formato HH:MM para mostrarla de manera clara.
8. Devuelve una tupla con la hora en formato de texto y el saludo correspondiente.

Propósito:
saludo mejora la experiencia del usuario en la aplicación al personalizar el mensaje de bienvenida según el momento del día, creando una interacción más cálida y cercana.

4.4.5. loading_animation

Descripción:
Esta función genera una animación de carga en la consola utilizando una secuencia de emojis de gatos y puntos, simulando una espera mientras el sistema procesa alguna tarea.

Funcionamiento:

1. Define un texto base de carga, "Espere un momento", y una lista de emojis (animation_frames) para crear los diferentes estados de la animación.
2. Ejecuta la animación en un bucle. En cada iteración:
3. Añade puntos a loading_text para simular el progreso.
4. Limpia la consola con clear_output(wait=True) para actualizar la animación en el mismo lugar de la pantalla.
5. Muestra el texto de carga y un emoji de la lista animation_frames.
6. Pausa brevemente usando time.sleep(0.3) para que la animación sea visible.

7. Tras dos ciclos completos, muestra el mensaje "¡Carga completa!" para indicar que el proceso de espera ha finalizado.

Propósito:
loading_animation añade un toque visual atractivo y entretenido para mejorar la experiencia del usuario durante los tiempos de espera, haciendo la interfaz más amigable y dinámica.

4.4.6. activitySearch

Descripción:
Esta función busca actividades en una base de datos según la similitud semántica con el texto de entrada (prompt). Utiliza generados por un modelo de transformers para identificar las actividades más relevantes.

Parámetros:

- prompt (str): Frase de entrada proporcionada por el usuario, que describe la actividad que está buscando.
- modelo: Modelo de *embeddings* que se utiliza para codificar el prompt y calcular las similitudes (en este caso, se hace referencia a model2, aunque el parámetro debería ser modelo).
- db (DataFrame): Base de datos que contiene las actividades disponibles, junto con sus *embeddings* pre-calculados en una columna llamada tensor.
- quantity (int): Número de resultados que se desean retornar, limitando así la cantidad de actividades mostradas en la salida.

Funcionamiento:

1. Genera un *embedding* del texto ingresado (prompt) usando modelo.encode para convertirlo en un tensor, lo que facilita la comparación semántica.
2. Calcula la similitud entre este *embedding* de entrada y cada *embedding* de actividad previamente almacenado en la columna tensor del dataframe db, utilizando util.cos_sim.
3. Añade una nueva columna, coincidencia_sbert, en el dataframe para guardar la similitud calculada para cada actividad.
4. Ordena el dataframe de mayor a menor similitud y selecciona las primeras quantity actividades relevantes.
5. Muestra los resultados, mostrando el título, la descripción y el valor de similitud para cada una de las actividades recomendadas.

Propósito:
activitySearch permite al usuario encontrar actividades que se alineen con sus intereses de manera rápida y precisa. Al ordenar las actividades por similitud, ayuda a que el sistema sugiera las opciones más relevantes según el contenido del prompt ingresado.

4.4.7. respuestaEmocion

Descripción:
Esta función maneja la respuesta del sistema según la emoción determinada del usuario y si se detectó alguna frase de riesgo (red_flag). Basándose en la emoción del usuario, ofrece recomendaciones de actividades que pueden mejorar su estado de ánimo.

Parámetros:

- emocion_determinada (int): Valor que representa la emoción del usuario, donde:
  - -1 indica que el usuario está triste.
  - 0 indica que el usuario tiene un estado emocional neutro.
  - 1 indica que el usuario está feliz.
- red_flag (bool, opcional): Indica si se ha detectado un patrón de lenguaje potencialmente peligroso. Por defecto, es False.

Funcionamiento:

- **Estado Neutro (emocion_determinada == 0):**
  1. Pregunta al usuario si desea recibir una recomendación de actividades.
  2. Si el usuario responde afirmativamente, le ofrece elegir entre películas, juegos de mesa, libros o salir.
  3. Según la elección del usuario, se le pide que indique qué le gustaría ver o jugar, y se llama a la función activitySearch para proporcionar las recomendaciones pertinentes.
- **Estado Triste (emocion_determinada == -1):**
  1. Si se detecta un red_flag, se recuerda al usuario que es importante hablar con alguien antes de tomar decisiones drásticas.
  2. Si no hay red flag, se ofrece al usuario la posibilidad de ver una película para intentar alegrar su día. Si acepta, se le pide que indique qué tipo de película le gustaría ver y se llama a activitySearch para ofrecer recomendaciones.
- **Estado Feliz (emocion_determinada == 1):**
  1. Se felicita al usuario por su estado emocional positivo y se le invita a compartir algo que le inspire. Luego se le pregunta si desea recibir recomendaciones.
  2. Si acepta, se realizan tres búsquedas a través de activitySearch para obtener las tres mejores coincidencias en películas, juegos de mesa y libros según lo que el usuario desea ver.

Propósito:
La función respuestaEmocion busca involucrar al usuario en una interacción significativa basada en su estado emocional, ofreciendo alternativas que puedan mejorar su ánimo o simplemente entreteniéndolo, y también prioriza la seguridad al reconocer situaciones de riesgo.

4.4.8. menu

Descripción:
Esta función presenta el menú principal del sistema WILSON, que está diseñado para interactuar con el usuario, recibir su estado emocional y proporcionar recomendaciones de actividades.

Parámetros:

- intro (bool, opcional): Determina si se debe mostrar una introducción al inicio del menú. Por defecto, es True.

Funcionamiento:

1. Introducción (intro=True):
    - Llama a la función loading_animation() para mostrar una animación de carga.
    - Limpia la salida de la consola para una mejor presentación.
    - Muestra el título del proyecto y los nombres de los integrantes con breves pausas entre cada impresión.
    - Presenta un mensaje de bienvenida al sistema WILSON y realiza otra limpieza de la consola.
2. Saludo y Consulta de Emoción:
    - Llama a la función saludo() para obtener la hora actual y un saludo apropiado según el momento del día.
    - Imprime la hora actual y un mensaje de introducción del sistema WILSON.
    - Solicita al usuario que ingrese cómo se siente en ese momento.
3. Análisis y Respuesta:
    - Limpia nuevamente la salida de la consola.
    - Llama a analisisEmocion() para analizar la emoción del usuario utilizando el modelo de aprendizaje automático y recibir la emoción determinada.
    - Llama a redFlagIdentifier() para identificar si la entrada del usuario contiene frases de riesgo.
    - Pasa los resultados de ambos análisis a la función respuestaEmocion() para manejar la respuesta basada en el estado emocional del usuario.

Propósito:
La función menu sirve como el punto de entrada interactivo para el sistema WILSON, guiando al usuario a través de una experiencia amigable y facilitando un análisis emocional que permite proporcionar recomendaciones personalizadas.

Se asegura de que la experiencia comience de manera cálida y acogedora, al tiempo que prioriza la claridad y la eficacia en la interacción.

4.4.9. start

Descripción:
Esta función inicia el sistema WILSON llamando a la función menu() con el parámetro intro establecido en True.

Funcionamiento:

- Simplemente invoca la función menu(intro), que se encarga de presentar la interfaz inicial del sistema, mostrando la introducción, el saludo, y solicitando la entrada del usuario sobre su estado emocional.

Propósito:
La función start actúa como un punto de partida sencillo para el programa. Su único propósito es facilitar la llamada al menú principal, lo que permite al usuario comenzar la interacción con el sistema de manera fluida y directa.

4.5. Carga de Base de datos

4.5.1. Base de datos para determinar emociones

En esta sección, se describe el proceso de carga y transformación de la base de datos utilizada para identificar las emociones del usuario. La información se extrae de un archivo de Excel que contiene una lista de emociones etiquetadas.

Primero, se carga el contenido del archivo en una estructura de datos llamada DataFrame. Esta estructura permite un manejo eficiente de los datos, facilitando su análisis y manipulación.

Para que el modelo de análisis de emociones pueda trabajar correctamente, es necesario transformar estas etiquetas textuales en valores numéricos. Esto se realiza mediante un proceso de mapeo que asigna un número específico a cada tipo de emoción. Por ejemplo, las emociones se categorizan de la siguiente manera:

- La emoción de "triste" se convierte en un valor de -1.
- La emoción "neutro" se asigna el valor 0.
- La emoción "feliz" se representa con el valor 1.

Este mapeo es fundamental porque permite que el modelo entienda y procese las emociones de manera adecuada, ya que los modelos de aprendizaje automático operan principalmente con valores numéricos.

4.5.2. Base de datos de juegos, películas y libros

4.5.2.1. Películas

La base de datos se extrae desde un archivo CSV utilizando la librería de pandas. No se le realiza ninguna transformación luego.

4.5.2.2. Juegos de mesa

En está caso también se extrae los datos desde un archivo CSV utilizando la librería de pandas y se renombra las columnas “game_name” por “Title” y “description” por “Description” para tener consistencia en todos los dataset utilizados en éste trabajo.

4.5.2.2. Libros

Para los libros no contamos con un archivo XLS o CSV, en este caso se tuvo que extraer la información mediante una página web utilizando técnicas de web scraping.

El primer paso consiste en acceder a la URL que presenta el listado de los 1000 libros más populares en la plataforma. Utilizando una solicitud HTTP, se verifica que la respuesta sea exitosa. En caso afirmativo, se procede a extraer el contenido de la página utilizando la biblioteca BeautifulSoup, que facilita el análisis y la manipulación de documentos HTML.

Una vez que se obtiene el contenido, se busca la sección específica donde se encuentran los títulos de los libros. A través de selectores CSS, se identifican y seleccionan todos los enlaces que llevan a las páginas individuales de cada libro.

Con una lista de enlaces a los libros, se itera sobre cada uno de ellos para recopilar información adicional. Para cada libro, se construye una URL que permite acceder a su página específica en el sitio. Nuevamente, se realiza una solicitud para obtener el contenido de esta nueva página.

En esta página de libro, se busca una tabla que contiene información relevante, como el título y un resumen del libro. Se utilizan encabezados específicos para localizar estos datos, y se extraen tanto el título como el resumen, que se almacenan en una lista de diccionarios.

Finalmente, toda la información recopilada se convierte en un DataFrame de pandas, lo que facilita su análisis y manipulación en etapas posteriores del proyecto. Este conjunto de datos es valioso, ya que permite enriquecer la

experiencia del usuario al proporcionar recomendaciones personalizadas de libros basadas en sus preferencias y emociones.

Este proceso de web scraping es esencial, ya que permite obtener datos de manera automatizada y efectiva, asegurando que el conjunto de datos esté siempre actualizado y sea representativo de las tendencias literarias actuales.

5. Resultados

5.1. Detección de emociones

Nuestro modelo de detección de emociones fue inicialmente entrenado con una base de datos que contenía 50 elementos por cada tipo de emoción. Sin embargo, durante las pruebas observamos que en frases más cotidianas, especialmente aquellas que incluían lunfardo, el modelo no lograba identificar correctamente la emoción. Para mejorar su precisión y adaptabilidad, decidimos ampliar la base de datos a 302 elementos por emoción, permitiendo una mayor diversidad de expresiones y un mejor rendimiento en contextos coloquiales.

El modelo fue evaluado con el conjunto de prueba y se analizaron distintas métricas para analizar su desempeño, como se observa en la Figura 1.

Precisión Regresión Logística: 0.7472527472527473
Reporte de clasificación Regresión Logística:

	precision	recall	f1-score	support
-1	0.84	0.73	0.78	37
0	0.69	0.71	0.70	28
1	0.70	0.81	0.75	26
accuracy			0.75	91
macro avg	0.74	0.75	0.74	91
weighted avg	0.76	0.75	0.75	91

Figura 1: Reporte del modelo de clasificación.

Precision:

- El 84% de las instancias marcadas como “-1” fueron correctas.
- El 69% de las instancias marcadas como “0” fueron correctas.
- El 70% de las instancias marcadas como “1” fueron correctas.

Recall:

- El 73% de los verdaderos positivos de la clase “-1” fueron correctamente identificados.
- El 71% de los verdaderos positivos de la clase “0” fueron correctamente identificados.
- El 81% de los verdaderos positivos de la clase “1” fueron correctamente identificados.

f1-score:

- Este valor combina precisión y recall. Se obtuvo entre 0.70 y 0.78 para todas las clases, lo cual indica una buena capacidad del modelo para detectar cada clase.

Accuracy:

- El modelo clasifica correctamente el 75% de los datos de prueba.

5.2. Recomendación de actividad

Gracias al modelo utilizado, se pueden obtener las similitudes semánticas entre el prompt en español ingresado por el usuario y los datos en inglés de la base de datos.

El sistema está diseñado para adaptar las recomendaciones según el estado emocional del usuario, que fue detectado previamente. Las recomendaciones varían de la siguiente manera:

- **Cuando el usuario está triste**: se le recomienda directamente una película, buscando ofrecer entretenimiento sencillo y de poco esfuerzo, que pueda ayudar a mejorar su estado de ánimo. Se le ofrece una sola película para evitar que el usuario deba gastar esfuerzo en la elección entre varias opciones.
- **Cuando el usuario está neutral**: se le permite elegir el tipo de actividad que prefiere, ya sea una película, un libro o un juego de mesa.
- **Cuando el usuario está feliz**: primero se le solicita el prompt específico sobre lo que le gustaría ver o hacer, y luego se le presentan opciones variadas, incluyendo películas, libros y juegos de mesa, para darle mayor libertad en su elección.

5.3. Rendimiento del código

Con el objetivo de disminuir el tiempo de ejecución, se tomó la decisión de remover 2 procesos del código:

- *Web-scraping* para generar la base de datos de libros
- *Embedding* de las 3 bases de datos de recomendaciones.

Para disminuir el tiempo de ejecución del código final, se optó por extraer las bases de datos finales obtenidas luego del *web-scraping* y del *embedding*. Así, las bases de datos utilizadas en el código final ya poseen todas las modificaciones

necesarias al momento de la ejecución. De esta manera se logra evitar repetir este proceso cada vez que se decide ejecutar el código en una nueva sesión.

6. Conclusiones

6.1. Resumen de resultados y verificación de Objetivos

Nuestro sistema de recomendaciones, **WILSON**, logra identificar el estado de ánimo del usuario, con cierta capacidad de adaptación ante el uso de lunfardo y expresiones coloquiales. En base a este estado de ánimo, le ofrecerá al usuario una serie de opciones para ocupar su tiempo, las cuales luego serán finalmente seleccionadas en base al prompt ingresado por el usuario.

Podemos afirmar que WILSON cumple su función correctamente, ya que de manera muy simple, le ofrece al usuario una actividad en la cual ocupar su tiempo libre que se adapte a sus necesidades emocionales del momento.

6.2. Posibles mejoras

Durante la evaluación de la aplicación detectamos puntos donde se podría mejorar en próximas actualizaciones de **WILSON** dentro de ellos destacamos las siguientes:

- **Expansión del dataset** que contiene las frases de distintas emociones: si bien con 302 frases y palabras por emoción logramos un modelo que predice bien con buena probabilidad, se le podría agregar más frases teniendo en cuenta otros idiomas, lunfardo argentino, etc.
- **Traducción de las recomendaciones**: se podría implementar un modelo de traducción que localice las descripciones de los juegos, películas y libros recomendados.
- Incluir selección de películas, libros o juegos por categoría y no solo por descripción.
- **Expandir la gama de emociones:** se podrían agregar más emociones que las 3 principales para poder aproximar mejor al sentimiento que tiene el usuario y así WILSON podrá dar un acompañamiento más personalizado.

7. Anexos

Google Colab Notebook con el código desarrollado en este informe:
https://colab.research.google.com/drive/1vNmvyB5pXervKlgjZs3T4PEYLTmzjUwg?usp=sharing