Proyecto fin master Big Data y Machine Learning



Modelos transformers generativos

# 

[**Introducción 3**](#_tb0fnapnx6m7)

[**Breve sumario de productos o resultados obtenidos 3**](#_mpr5llulso5i)

[**Contexto y justificación del Trabajo 5**](#_tuitfd4an3y6)

[Objetivos del Trabajo. 5](#_lljteljufobh)

[Enfoque y metodología. 5](#_qebb0bz1vswp)

[Planificación del Trabajo. 6](#_8zy6dv8jrgph)

[Recursos de sistemas utilizados. 6](#_rdhqp5xq9puo)

[**Creación del set de datos 6**](#_z1oemzuzn0d7)

[Estudio de datos necesarios. 6](#_ax1mu2fivela)

[Fuente de datos. 7](#_24l69rgh8y1e)

[Recolección de los datos. 7](#_hiiriuowckji)

[Visualización de datos. 8](#_v4leyklzmsng)

[**Optimización, normalización y calidad del set de datos. 9**](#_r29e9843jg4p)

[**Desarrollo del modelo predictivo 11**](#_nx9x9fq50s2u)

[Justificación del modelo seleccionado. 16](#_lzfh998z38ab)

[Descripción del modelo. 17](#_4troz1k4zcha)

[**Entrenamiento y evaluación del modelo predictivo. 19**](#_2d3sdv5ukav)

[**Informe de rendimiento y métricas en aula virtualizada y otros entornos. 21**](#_he6v24bdkfzd)

[**Puesta en producción (API y aplicación cliente). 22**](#_gv4613z5mwi9)

[**Conclusiones y mejoras a realizar, modelos alternativos. 23**](#_4d9t8q28j12t)

[Webgrafía 24](#_ncjz5hicicdw)

# 

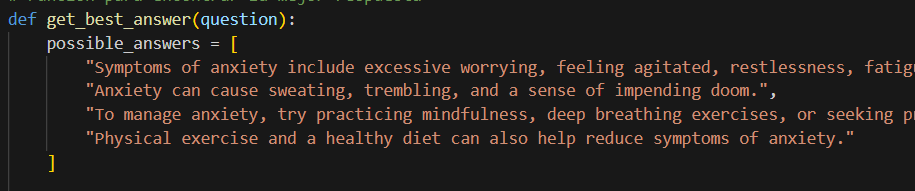
# Introducción

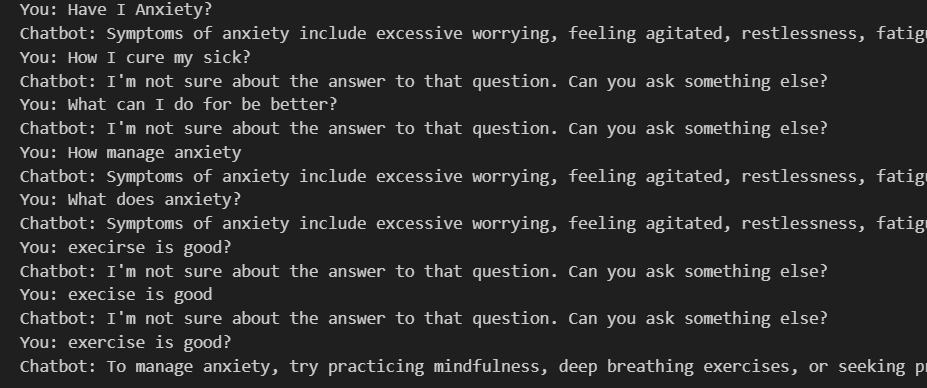
La inteligencia artificial (IA) ha revolucionado muchos campos, entre ellos el de la salud mental, ofreciendo nuevas herramientas para el diagnóstico, tratamiento y apoyo a los pacientes. En este contexto, los chatbots se han convertido en una solución innovadora para proporcionar información y apoyo a las personas que enfrentan problemas de salud mental. Este trabajo de fin de máster se centra en el desarrollo de un chatbot utilizando modelos de lenguaje natural, específicamente el modelo GPT-2, entrenado con un dataset de preguntas frecuentes sobre enfermedades mentales.

El objetivo principal de este proyecto es desarrollar un chatbot capaz de interactuar de manera efectiva y precisa con los usuarios, proporcionando respuestas basadas en información fiable y autorizada. Para ello, se ha utilizado el dataset Mental\_Health\_FAQ disponible en Hugging Face y se ha experimentado con diferentes modelos y técnicas de entrenamiento antes de llegar al modelo final.

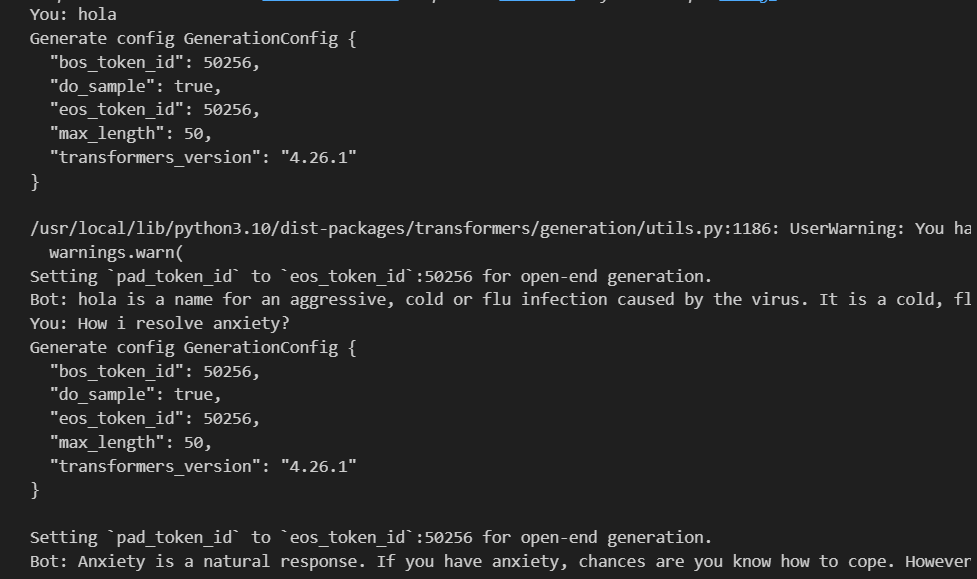
Este trabajo se estructura en varias secciones, comenzando con la descripción de los objetivos del proyecto, seguido por el enfoque y la metodología adoptada, el estudio de los datos necesarios, la fuente de datos, y el proceso de recolección de los mismos. Cada una de estas secciones detallará los pasos y las decisiones tomadas durante el desarrollo del chatbot, así como los desafíos encontrados y las soluciones implementadas.

# Breve sumario de productos o resultados obtenidos

El primer modelo generado con finetunning a partir de bert\_base\_uncased, responde a la respuesta más probable a tu pregunta con una bateria de respuestas ya guardadas:



El segundo modelo generado con finetunning a partir de GPT2 podía mantener una conversación pero muy centrada en el contexto de su entrenamiento:



El tercer modelo generado con finetunning a partir de GPT2 incluyendo adapters podía mantener una conversación fluida, lo entrené con un libro en castellano para que aprendiera español el resultado es que cuando le preguntabas cuestiones del libro de meditaciones te respondía en una especie de protoespañol:











# Contexto y justificación del Trabajo

## Objetivos del Trabajo.

Desarrollar un chatbot basado en GPT-2 capaz de responder a preguntas frecuentes sobre enfermedades mentales, utilizando el dataset Mental\_Health\_FAQ.

Explorar y comparar diferentes modelos y técnicas de procesamiento de lenguaje natural (NLP) para determinar la más adecuada para este propósito.

Evaluar la eficacia y precisión del chatbot mediante pruebas y retroalimentación de usuarios.

Crear un segundo chatbot entrenado con el libro 'Meditaciones' de Marco Aurelio, ofreciendo respuestas filosóficas y reflexivas a las preguntas de los usuarios.

Documentar todo el proceso de desarrollo, desde la selección de datos y modelos hasta la implementación y evaluación final del chatbot.

## Enfoque y metodología.

El desarrollo del chatbot siguió un enfoque iterativo y experimental, en el cual se probaron múltiples modelos y técnicas antes de llegar a la solución final. El proceso se dividió en las siguientes etapas:

Revisión de Literatura y Análisis del Dataset: Se inició con una revisión exhaustiva de la literatura existente sobre chatbots en salud mental y procesamiento de lenguaje natural. El dataset Mental\_Health\_FAQ fue analizado para comprender su estructura y contenido.

Selección de Modelos: Se probaron varios modelos de NLP, comenzando con BERT base uncased y BertForSequenceClassification. Este modelo seleccionaba la respuesta más adecuada a cada pregunta, pero no era suficiente para la complejidad del tema.

Pruebas con Modelos Alternativos: Se exploraron otros modelos como LLaMA3, que ocupaba más de 250GB de disco al entrenar, lo cual resultó inviable. Posteriormente, se probó con LLaMA2, pero también se descartó debido a su obsolescencia.

Entrenamiento con GPT-2 y Adapters: Finalmente, se decidió entrenar el modelo GPT-2 con adapters, lo que permitió crear un chatbot más eficiente y adecuado para las necesidades del proyecto. Los adapters son módulos que permiten ajustar modelos preentrenados a nuevas tareas sin necesidad de entrenarlos desde cero.

Desarrollo del Segundo Chatbot: Paralelamente, se desarrolló un segundo chatbot utilizando el libro 'Meditaciones' de Marco Aurelio, para ofrecer una perspectiva diferente y reflexiva en las respuestas.

## Planificación del Trabajo.

## Recursos de sistemas utilizados.

Google Colab Pro.

GPU NVIDIA Tesla T4, especificaciones de reloj:

Frecuencia base del núcleo (Base Clock): 585 MHz

Frecuencia de aumento del núcleo (Boost Clock): 1590 MHz

Frecuencia de la memoria: 5000 MHz (efectiva)

La NVIDIA Tesla T4 utiliza memoria GDDR6, y la velocidad de reloj efectiva de la memoria (RAM) es de 10000 MHz. Esta alta frecuencia es crucial para proporcionar el ancho de banda necesario para las aplicaciones de aprendizaje profundo y computación intensiva. Cuenta con 51GB de RAM.

Almacenamiento de 256GB Nvme 5400MB/S

# Creación del set de datos

## Estudio de datos necesarios.

El éxito de un chatbot depende en gran medida de la calidad y relevancia de los datos utilizados para su entrenamiento. Para este proyecto, se identificaron dos conjuntos de datos principales:

Dataset Mental\_Health\_FAQ: Este dataset incluye una amplia gama de preguntas y respuestas relacionadas con enfermedades mentales, proporcionando una base sólida para el entrenamiento del chatbot en este dominio. Se seleccionaron datos que abarcaban diversas enfermedades, síntomas, tratamientos y estrategias de manejo.

Libro 'Meditaciones' de Marco Aurelio: Para el segundo chatbot, se utilizó el texto completo del libro 'Meditaciones', un clásico de la filosofía estoica. Este conjunto de datos se seleccionó para ofrecer respuestas basadas en principios filosóficos, proporcionando una perspectiva reflexiva y orientada al bienestar emocional.

Ambos conjuntos de datos fueron preprocesados para asegurar su compatibilidad con los modelos de NLP utilizados. Esto incluyó la limpieza de texto, la tokenización y la normalización de los datos.

## Fuente de datos.

Dataset Mental\_Health\_FAQ: Disponible en Hugging Face, este dataset fue creado por expertos en salud mental y se actualiza regularmente para incluir la información más reciente y relevante. Se verificó la validez del dataset a través de la revisión de la literatura y la consulta con profesionales del área.

URL de descarga: https://huggingface.co/datasets/tolu07/Mental\_Health\_FAQ



Libro 'Meditaciones' de Marco Aurelio: Se utilizó una versión digital del libro obtenida de fuentes de dominio público. La validez de esta fuente está respaldada por la reputación del texto como un clásico de la filosofía y su disponibilidad en múltiples ediciones revisadas.

URL de descarga: https://www.elejandria.com/libro/link\_descarga\_libro/1301/4153

## Recolección de los datos.

La recolección de datos para este proyecto implicó varios pasos críticos:

Descarga y Preprocesamiento del Dataset Mental\_Health\_FAQ: Se descargó el dataset de Hugging Face y se realizó un preprocesamiento exhaustivo para limpiar y estructurar los datos. Esto incluyó la eliminación de duplicados, corrección de errores tipográficos y normalización del texto.

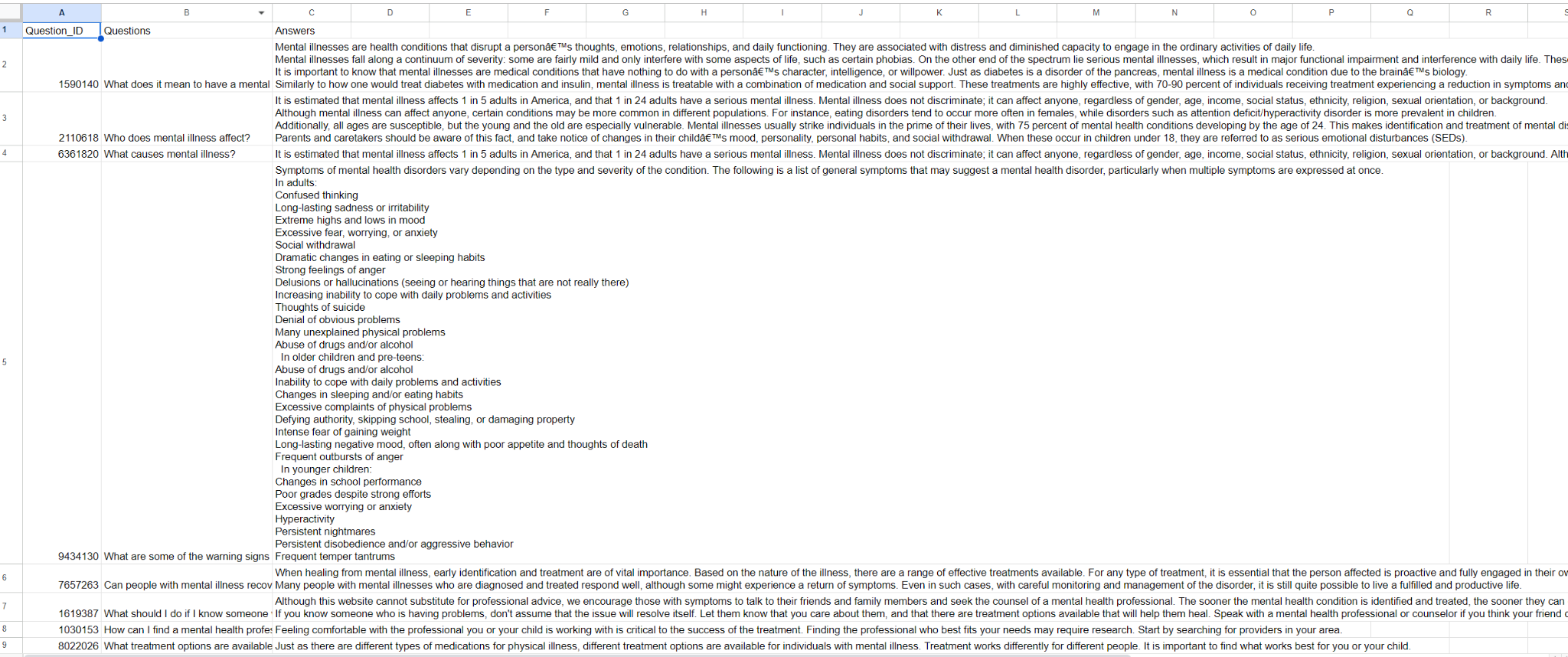
Digitalización y Preprocesamiento del Libro 'Meditaciones': El libro se descargó en PDF y se transformó a string, segmentando el contenido en unidades manejables para el entrenamiento del modelo a modo de párrafo.

Integración y Formateo de Datos: Ambos conjuntos de datos se integraron en un formato adecuado para el entrenamiento de los modelos de NLP. Se utilizó una tokenización adecuada para GPT-2, asegurando que el modelo pudiera manejar eficientemente los datos durante el entrenamiento.

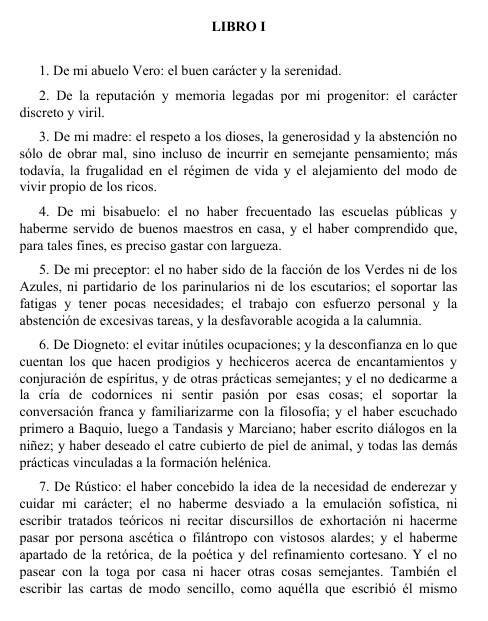
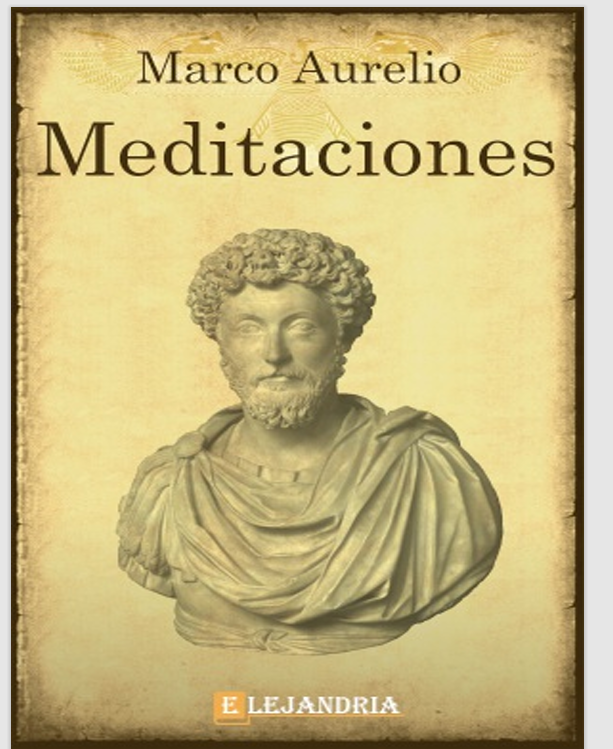
Desafíos Encontrados: El mayor desafío encontrado fue la necesidad de ajustes constantes en los parámetros de entrenamiento para mejorar el desempeño del modelo.

## Visualización de datos.

Para visualizar el dataset de las faq acerca de la salud mental use una herramienta de hoja de cálculo importando el csv para revisar la calidad de los datos facilitando la lectura:

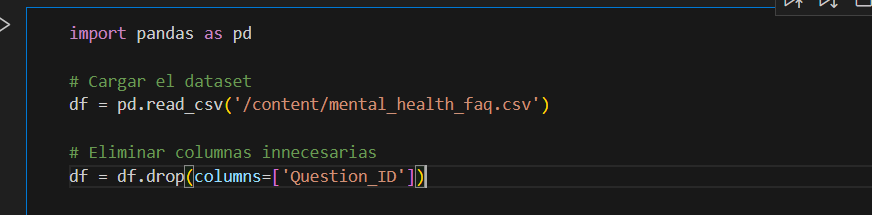


En cuanto al PDF de Meditaciones de Marco Aurelio al ser un PDF bastó con usar un lector de PDF:

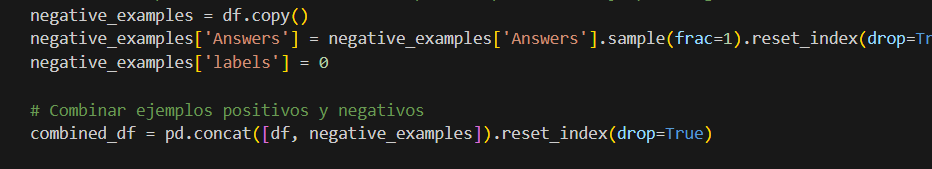


# Optimización, normalización y calidad del set de datos.

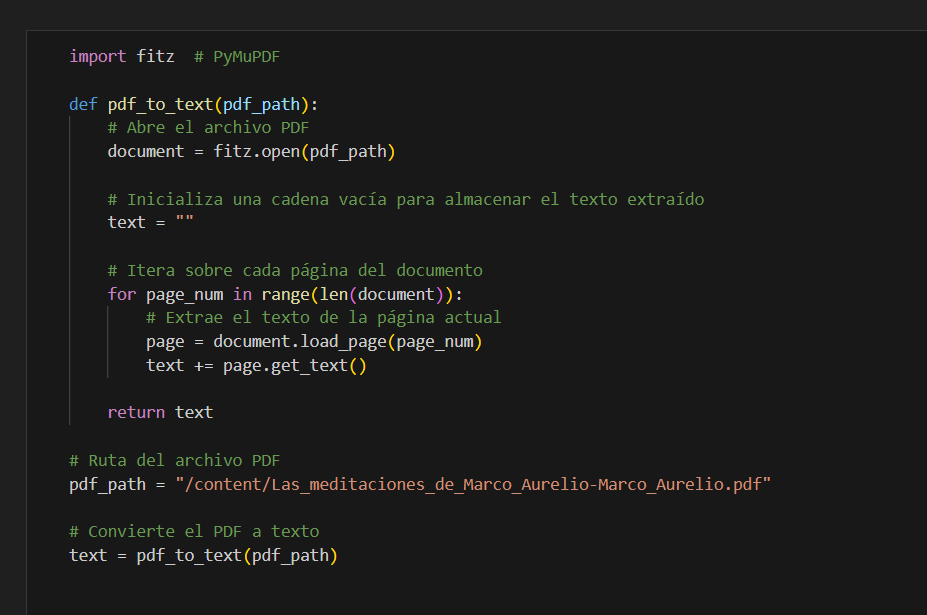
La normalización del dataset de faq sobre salud mental fue eliminar las columnas que no aportaban valor al análisis ni al entrenamiento del modelo. En este caso, se decidió eliminar la columna Question\_ID, ya que su única función era servir como identificador único de cada pregunta, lo cual no es relevante para el proceso de generación de respuestas del chatbot.



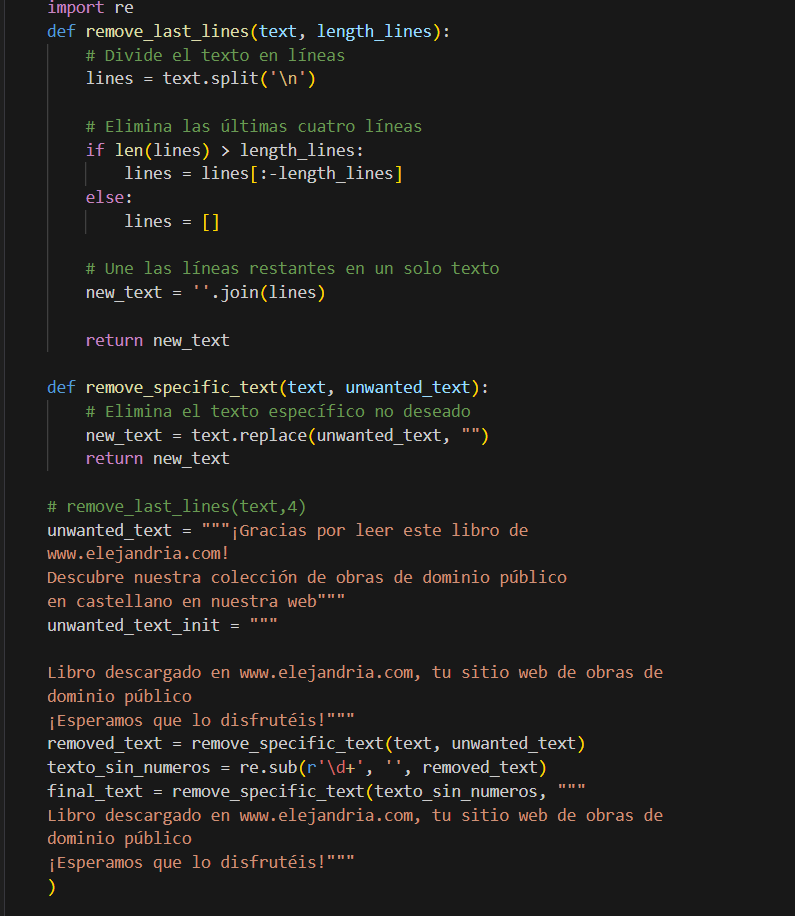
También se crearon las respuesta incorrectas desordenando las respuestas para crear ejemplos negativos y unirlos finalmente:



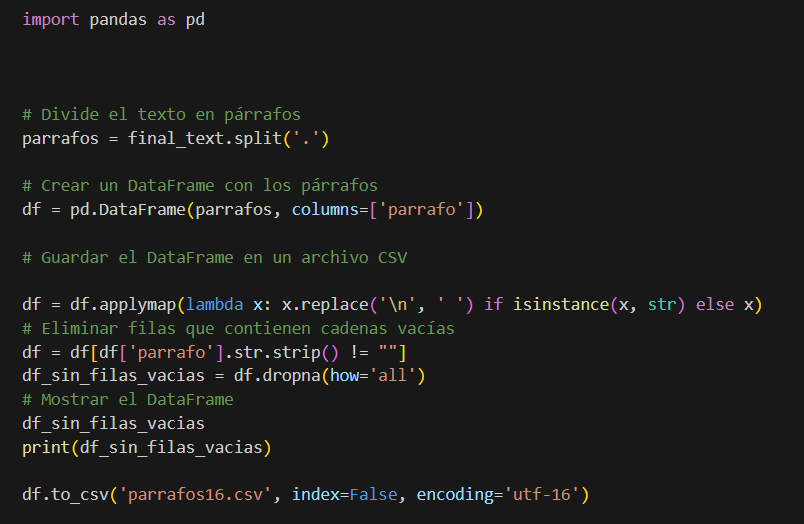
Importar el libro de meditaciones de Marco aurelio para entrenar el modelo fue más complejo tuve que usar una librería que lee PDFs para pasar el libro entero a string:



Una vez cargado el libro como una string hice una limpieza de los datos y elimine los números dado que no me parecían datos interesantes para entrenar al chatbot:



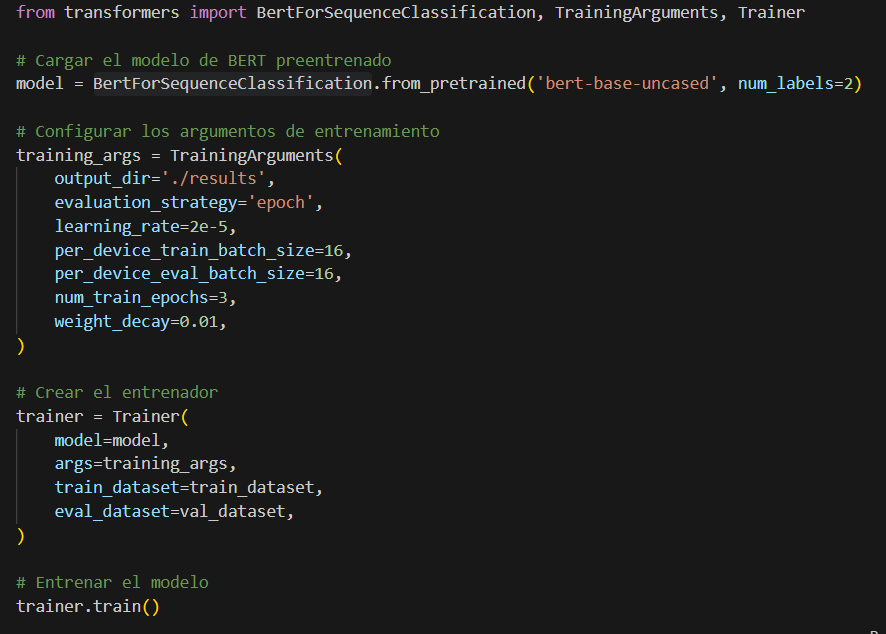
Para entrenar un modelo hay que pasarle un dataset por lo que decidí dividir la string por párrafos:



# Desarrollo del modelo predictivo

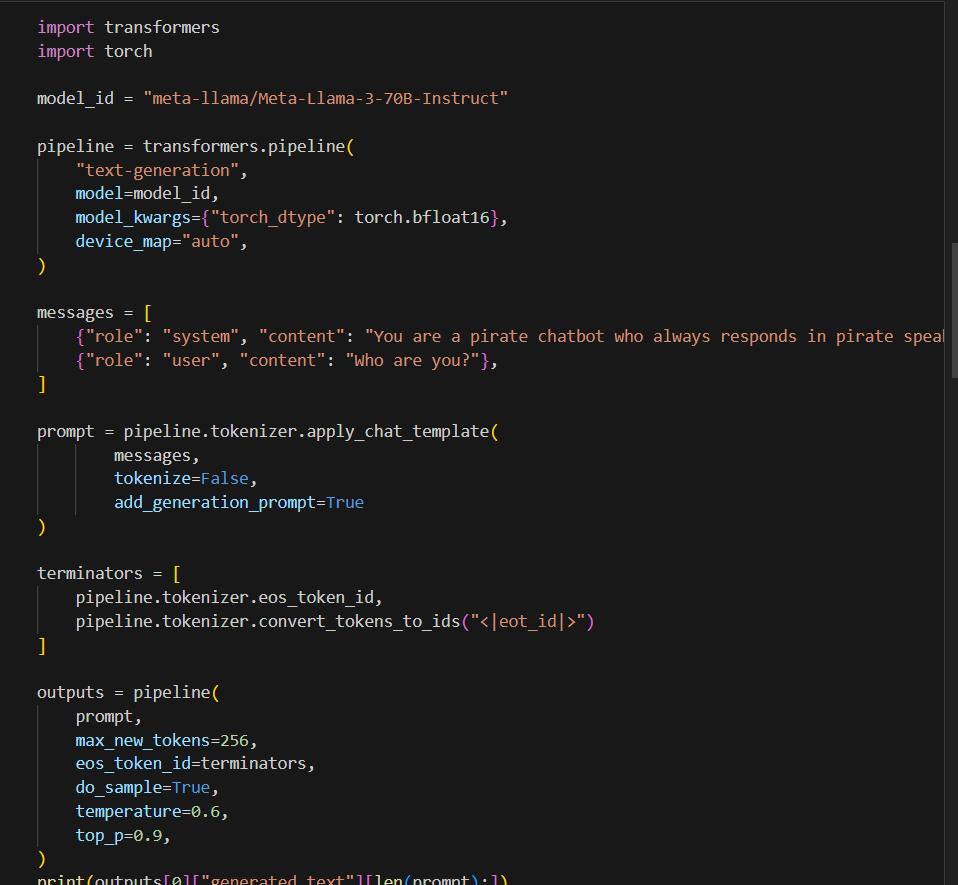
Primeros Intentos con BERT

Inicialmente, se utilizó el modelo BERT base uncased junto con la clase BertForSequenceClassification. La idea era aprovechar la capacidad de BERT para comprender y clasificar secuencias de texto. Se entrenó el modelo para seleccionar la respuesta más adecuada a una pregunta dada del dataset de FAQ. Sin embargo, los resultados no fueron satisfactorios, ya que el modelo no lograba generar respuestas coherentes y contextualmente adecuadas, limitándose a seleccionar respuestas predefinidas.



Intento con LLaMA 3

En búsqueda de un mejor rendimiento, se probó el modelo LLaMA 3, conocido por su capacidad para manejar tareas de generación de texto. No obstante, este modelo presentó un gran inconveniente: su entrenamiento requería más de 250 GB de espacio en disco, lo cual excedía considerablemente los recursos disponibles.

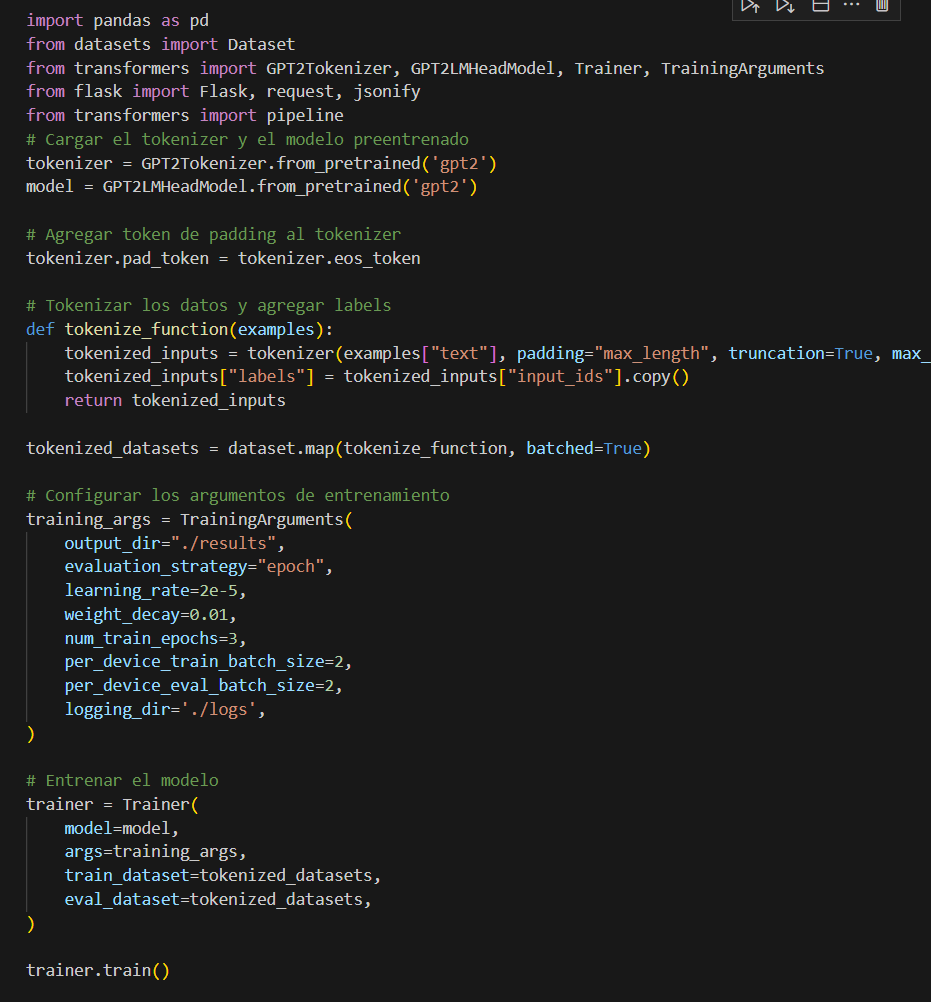


Cambio a LLaMA 2

Ante la imposibilidad de utilizar LLaMA 3, se consideró la versión anterior, LLaMA 2. Sin embargo, se descubrió que este modelo estaba deprecado y no ofrecía el soporte necesario.



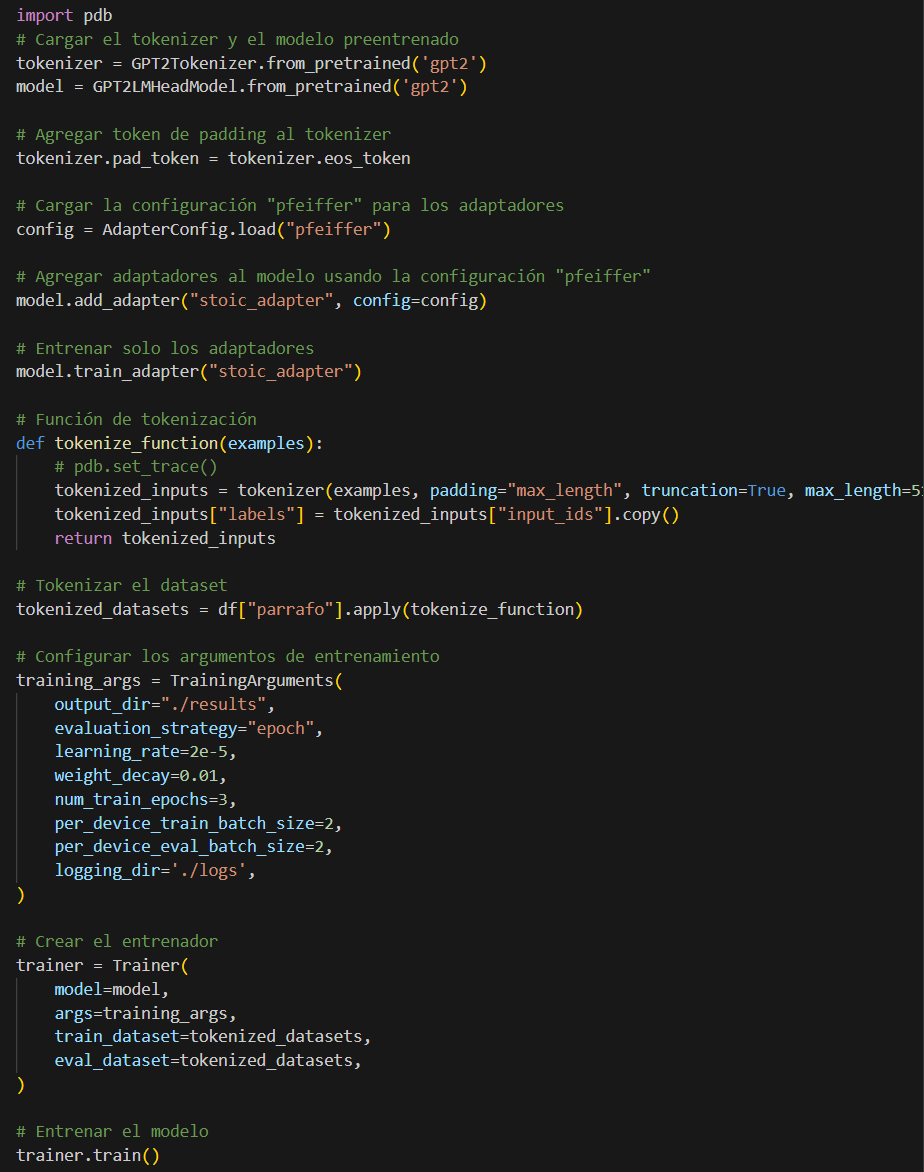
Entrenamiento con GPT2, fuel que mejor resultado me dió pero no estaba contento del todo con el nivel conversacional que tenía:



## 

## Justificación del modelo seleccionado.

Finalmente, se decidió utilizar el modelo GPT-2 debido a su capacidad comprobada para la generación de texto coherente y contextual. Además, el uso de adapters permitió adaptar el modelo a las necesidades específicas de los chatbots sin requerir recursos excesivos.



## Descripción del modelo.

Importación y Preparación del Modelo GPT-2:

python

Copiar código

tokenizer = GPT2Tokenizer.from\_pretrained('gpt2')

model = GPT2LMHeadModel.from\_pretrained('gpt2')

Se cargan el tokenizador y el modelo preentrenado GPT-2 de la librería transformers. GPT-2 es un modelo de lenguaje generativo capaz de generar texto coherente y relevante basado en el contexto proporcionado.

Token de Padding:

python

Copiar código

tokenizer.pad\_token = tokenizer.eos\_token

Se establece el token de padding para el tokenizador, igualándolo al token de fin de secuencia (eos\_token). Esto es importante para asegurar que las secuencias tokenizadas tengan la misma longitud durante el entrenamiento.

Carga y Configuración de Adaptadores:

python

Copiar código

config = AdapterConfig.load("pfeiffer")

model.add\_adapter("stoic\_adapter", config=config)

model.train\_adapter("stoic\_adapter")

Se cargan y configuran adaptadores usando la configuración "pfeiffer". Los adaptadores son pequeños módulos añadidos al modelo que permiten ajustar solo ciertas partes del modelo en lugar de todo el modelo. Esto reduce significativamente el costo computacional y la cantidad de datos necesarios para el ajuste fino.

Función de Tokenización:

def tokenize\_function(examples):

tokenized\_inputs = tokenizer(examples, padding="max\_length", truncation=True, max\_length=512)

tokenized\_inputs["labels"] = tokenized\_inputs["input\_ids"].copy()

return tokenized\_inputs

tokenized\_datasets = df["parrafo"].apply(tokenize\_function)

La función tokenize\_function se encarga de tokenizar el dataset, asegurando que cada ejemplo esté adecuadamente truncado y padding al máximo de longitud permitida (512 tokens en este caso).

Configuración de Argumentos de Entrenamiento:

training\_args = TrainingArguments(

output\_dir="./results",

evaluation\_strategy="epoch",

learning\_rate=2e-5,

weight\_decay=0.01,

num\_train\_epochs=3,

per\_device\_train\_batch\_size=2,

per\_device\_eval\_batch\_size=2,

logging\_dir='./logs',

)

Se configuran los argumentos para el entrenamiento, incluyendo la tasa de aprendizaje, el número de épocas, y el tamaño del lote por dispositivo.

Creación y Entrenamiento del Entrenador:

trainer = Trainer(

model=model,

args=training\_args,

train\_dataset=tokenized\_datasets,

eval\_dataset=tokenized\_datasets,

)

trainer.train()

Se crea un objeto Trainer que maneja el proceso de entrenamiento del modelo utilizando los datasets tokenizados y los argumentos de entrenamiento configurados.

Importancia del Uso de Adaptadores

El uso de adaptadores es crucial en este contexto por varias razones:

Eficiencia Computacional: Los adaptadores permiten ajustar solo una parte del modelo en lugar de todo el modelo, lo cual es menos costoso en términos computacionales y de memoria.

Reutilización de Modelos Preentrenados: Los adaptadores facilitan el uso de modelos preentrenados, añadiendo capacidades específicas sin alterar el modelo base.

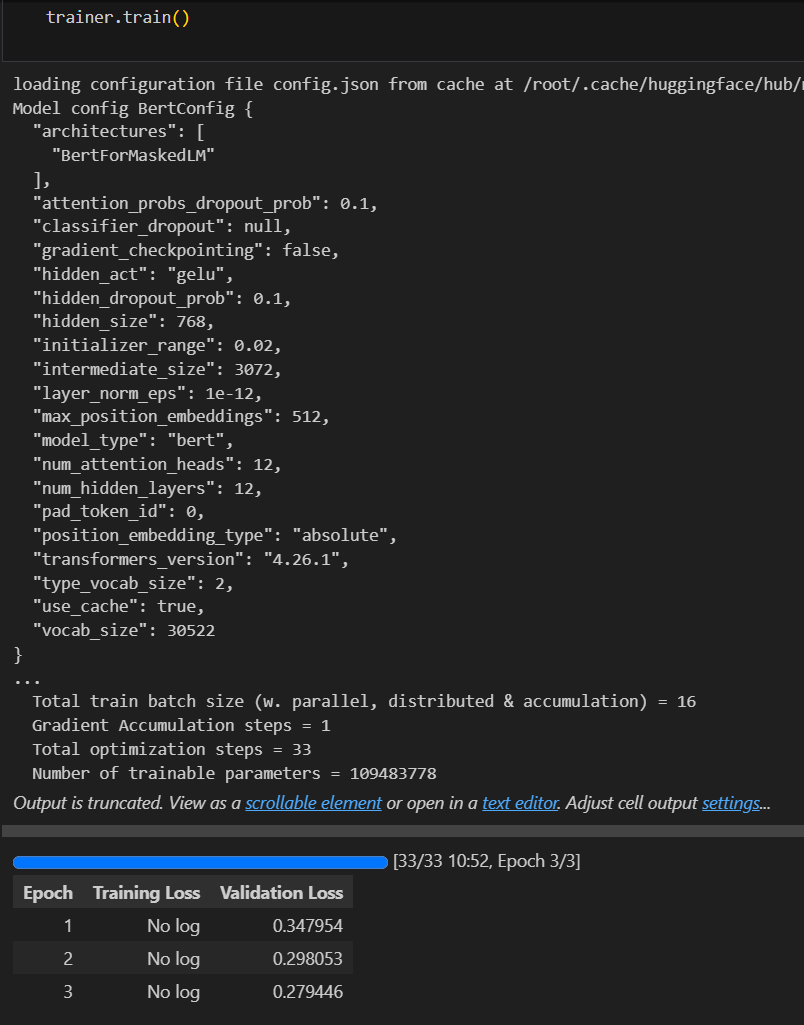
Flexibilidad: Es más sencillo intercambiar y probar diferentes configuraciones de adaptadores para ver cuál funciona mejor para un conjunto de datos o tarea específicos.

Menor Necesidad de Datos: Dado que solo se ajustan partes del modelo, se necesita menos cantidad de datos para lograr un ajuste fino efectivo.

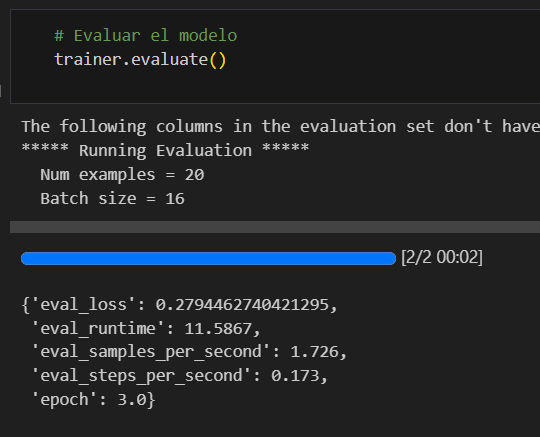
En resumen, este modelo utiliza el ajuste fino de GPT-2 con adaptadores para permitir respuestas generativas, beneficiándose de la eficiencia y flexibilidad que ofrecen los adaptadores.

# Entrenamiento y evaluación del modelo predictivo.

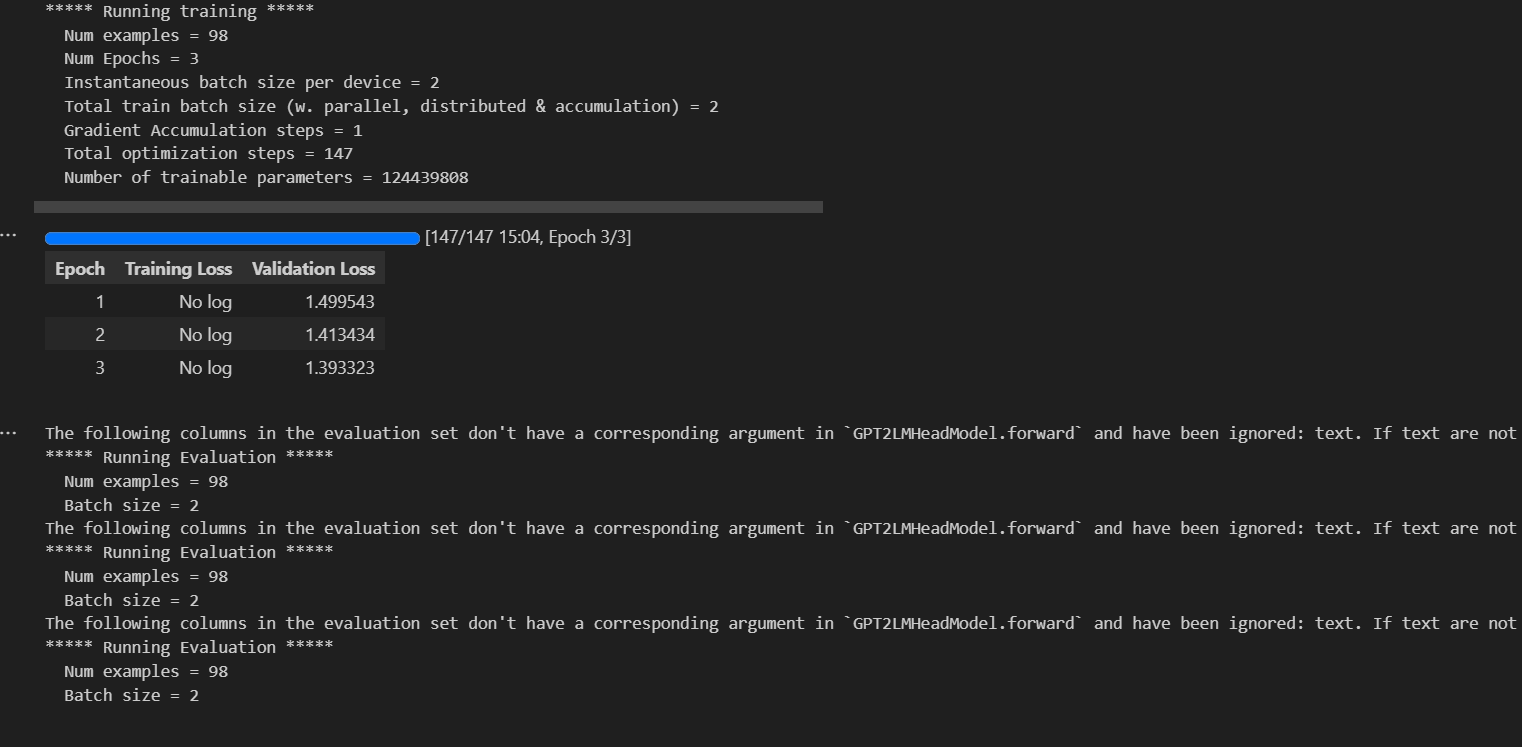
Entrenamiento Bot FAQ Bert:



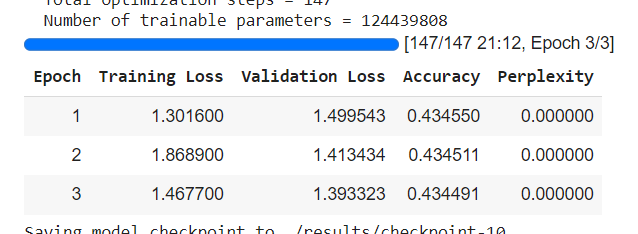
Entrenamiento GPT2 con adapter:



Entrenamiento bot FAQ GPT2:

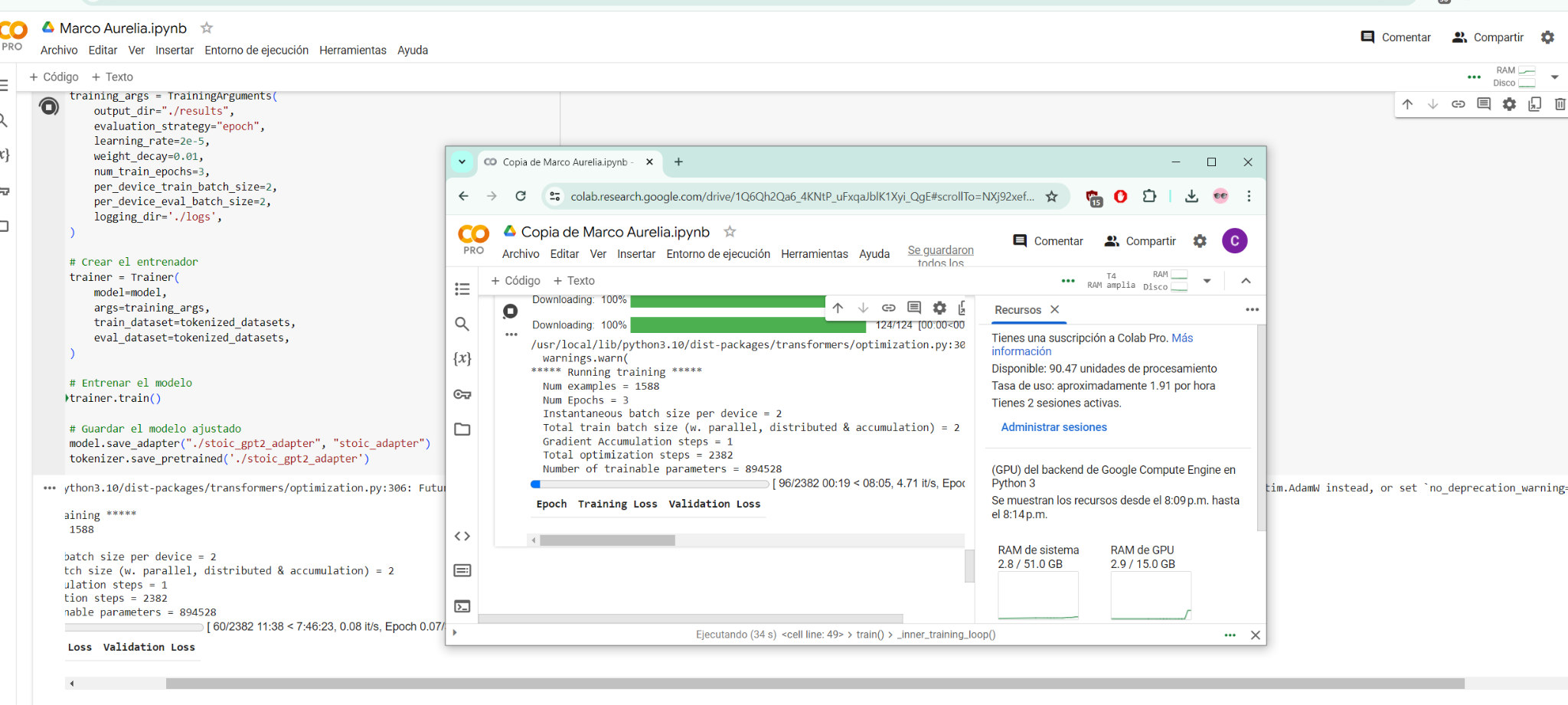


Validación del modelo predictivo:



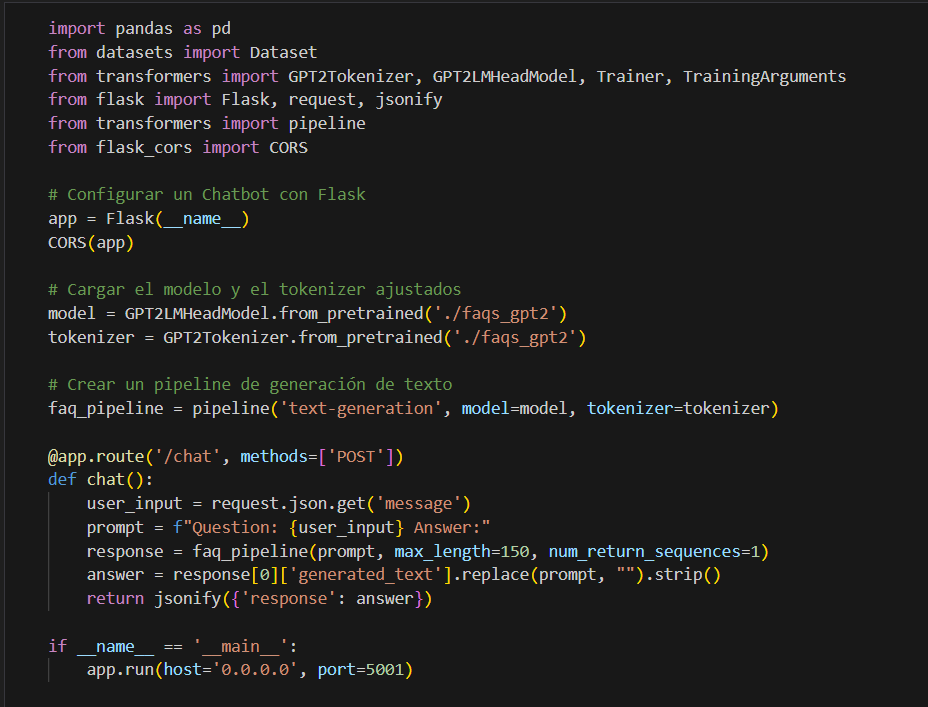
# Informe de rendimiento y métricas en aula virtualizada y otros entornos.

Tras probar con los entornos de colab y colab pro, se puede ver una diferencia de rendimiento abismal, mientras que con colab normal cada época requería de un entrenamiento de 7 horas y 46 minutos, colab pro realizaba el entrenamiento en 8 minutos, esto supone un rendimiento a la de entrenar de colab pro frente a colab de x58



# Puesta en producción (API y aplicación cliente).

La aplicación funciona bajo una API Rest con Flask como backend donde tenemos el modelo precargado y responde a las peticiones POST en el endpoint chat a las preguntas que se le realicen:



Para el frontend se desarrollo un maquetado con HTML5 que consulta mediante peticiones asíncronas con con AJAX de Jquery al endpoint creado con Flask:



# Conclusiones y mejoras a realizar, modelos alternativos.

La implementación de un chatbot eficaz depende en gran medida de la elección del modelo subyacente. En el caso de los chatbots desarrollados con GPT-2, se lograron resultados satisfactorios, pero aún existen áreas de mejora. Un modelo LLaMA 3 más liviano y multilingüe podría ofrecer varias ventajas.

Optimización del Espacio en Disco

Uno de los mayores desafíos con el modelo LLaMA 3 fue su consumo excesivo de espacio en disco, superando los 250 GB. Un modelo LLaMA 3 más liviano estaría diseñado para ofrecer una arquitectura optimizada que reduce significativamente el tamaño del modelo sin comprometer su rendimiento. Esto permitiría realizar entrenamientos y despliegues en infraestructuras con limitaciones de almacenamiento, haciendo el proceso más accesible y eficiente.

Eficiencia en el Uso de Memoria y Procesamiento

Un modelo más liviano también implicaría una menor demanda de memoria RAM y capacidad de procesamiento. Esto permitiría entrenar y ejecutar el modelo en hardware menos potente, como servidores con recursos limitados o incluso dispositivos locales. La reducción en el uso de recursos también se traduciría en menores costos operativos y una mayor eficiencia energética.

Mejora en la Comprensión Contextual

Un modelo multilingüe también tiende a mejorar la comprensión contextual debido a su entrenamiento en una mayor diversidad de datos. Esto significa que el chatbot podría entender mejor matices y variaciones lingüísticas, ofreciendo respuestas más precisas y culturalmente relevantes.

## Webgrafía

Recursos sobre Cromatografía de Suelo:

Introducción a la Cromatografía de Suelo - Journal of Soil Science

https://www.sciencedirect.com/journal/journal-of-soil-science

Manual de Cromatografía de Suelo - Universidad de Buenos Aires

http://www.uba.ar

Fundamentos de la Cromatografía de Suelo - Universidad Autónoma de Madrid

https://www.uam.es

Cromatografía de Suelo: Técnicas y Aplicaciones - ResearchGate

https://www.researchgate.net

Explicación del Fine-Tuning en IA:

Fine-Tuning en Machine Learning - Towards Data Science

https://towardsdatascience.com/fine-tuning-in-machine-learning-3af2725174b2

Guía Completa sobre Fine-Tuning - Medium

https://medium.com

Tutorial de Fine-Tuning - TensorFlow

https://www.tensorflow.org/tutorials

Fine-Tuning: Teoría y Práctica - Coursera

https://www.coursera.org