Universidad de Los Andes

PROYECTO 1

Etapa 2

Inteligencia de Negocios

María del Pilar Villamil Giraldo

2022

INTEGRANTES

Andres Felipe Arias Russi - 201914996

Diego Granada Martínez - 201922383

Verónica Escobar Aristizábal - 201922107

Proyecto 1 – Etapa 2

Contenido

Introducción

Se decidió trabajar bajo el contexto de salud mental, y se desarrollara el proyecto alrededor del apoyo a la detección de intentos de suicidio a partir de información de Reddit a nivel de comunidades que sufren de depresión o han intentado suicidarse. El proyecto se divide en dos etapas. La primera etapa está relacionada a la construcción de modelos de datos analíticos, y el rol principal del equipo es el de científicos de datos. La segunda etapa está relacionada a el despliegue de estos modelos analíticos y el desarrollo de una aplicación, aquí el rol principal es el de un ingeniero de datos.

Contexto del Caso

El set de datos usado proviene originalmente de una colección de posts realizados a las comunidades de r/SuicideWatch y r/Depression. El autor del dataset menciona que los datos fueron recolectados por medio de PutshiftAPI. Los datos de SuicideWatch toma un recuento de todos los posts hechos a esa comunidad desde diciembre de 2008 a enero de 2021, mientras que los datos de r/Depression fueron recolectados desde las fechas de enero de 2009 a enero de 2021. Por lo tanto, ambos data sets recolectan aproximadamente 12 años de posts en total de cada una de las comunidades. Aunque este es el set de datos original para la investigación el set de datos base que se va a usar en el proyecto es el proporcionado por el equipo de Inteligencia de Negocios en un ZIP que contiene dos archivos uno el cual contiene un set de datos sin clasificar si son suicidal o non-suicidal mientras que otro ya tiene las clasificaciones hechas. Este set de datos contiene una mezcla de posts que pueden ser clasificados como *suicidal* y otros de temáticas no relacionadas y que pueden ser clasificados como *non-suicidal.*

Etapa 2. Automatización de analítica de textos

Descripción de aplicación

La aplicación tenía el objetivo de otorgar una interfaz simple para que los usuarios tuvieran la habilidad de verificar la propensión del autor de un mensaje en redes sociales a tener pensamientos suicidas. A pesar de conseguir un modelo efectivo, no sirve de mucho si no se puede usar de una forma simple y efectiva. Es por esto por lo que buscamos desarrollar una aplicación web básica que permita mandar un mensaje de texto, y que le muestre al usuario el resultado de nuestra predicción, informándole al usuario que hacer en ambos casos. Se utilizó una aplicación web para asegurar que sea accesible desde la mayor cantidad de plataformas posibles. El propuesto de la aplicación es de servir como una herramienta de apoyo a un actor humano, por lo tanto, tiene que ser amigable y lógica para su uso,

Automatización del proceso de preparación de datos

El proceso de preparación de datos tal como fue planteado para la etapa 1 fue hecho de tal forma que fuese fácil de usar en cuando se implementase en una aplicación. Originalmente el procesamiento de los datos fue hecho de una forma secuencial y progresiva, que hace más fácil entender el efecto de cada operación dentro de los datos, pero para implementar el modelo esto no era necesario por lo tanto a la hora de crear los modelos individuales se optó por hacer el procesamiento como una sola función unificada. Todo lo relacionado al modelo y procesamiento en el proyecto se puede encontrar dentro la carpeta app en el archivo de PredictionModel.py. El procesamiento de los datos e implemento en una función llamada process_text que unificaba todos los pasos de limpieza de los datos y procesamiento para prepararlos para ser usados para hacer NLP con cualquier algoritmo que quisiésemos usar, independiente si fuese Naive Bayes, SVM o Regresión logística. Esta función se llama dentro de la función de predict del prediction model para así procesar los datos que se le estén pasando antes de que se realice una predicción con el algoritmo. En si no se enriqueció el modelo, pero la forma en la que se implementó se volvió más versátil.

Automatización de la construcción del modelo

En la etapa uno se determinó que el modelo que más se adecuada a el objetivo del proyecto era el de regresión logística. Dada esta información se decidió persistir el modelo localmente desde el notebook donde fue desarrollado, lo cual produce un pkl que guarda la información sobre el modelo. Para facilitar el proceso de implementar el algoritmo en el proyecto y hacer más versátil el proceso de cambiar de algoritmo a futuro se decidió no implementar directamente el modelo dentro de la aplicación, más se decidió simplemente importar el pkl que persiste un modelo para así poder recrearlo on-demand. De esta forma es fácil realizar cambios y pruebas sobre el modelo y fácilmente implementarlo dentro del proyecto con solo exportar el pkl producido al proyecto y cambiar su referencia en la clase de PredictionModel.

Todo esto agiliza el proceso de actualización y optimización del modelo y como este se ve desde el punto de vista de la aplicación que el cliente estaría usando. Esto hace que cuando se encuentre un mejor algoritmo no se tenga que hacer de nuevo una implementación única a la aplicación si no que solo se puede exportar el modelo creado y reusar tal como sea deseado. A futuro lo ideal sería lograr que cuando se cree un nuevo pkl en el notebook este fuese exportado o sincronizado automáticamente con el modelo del proyecto, pero esto estaba fuera de nuestras capacidades de implementación.

Acceso del modelo por medio de API

El backend de la aplicación fue hecho utilizando FastAPI, una framework para crear APIs REST en Python. Se utilizó debido a que no se requería el uso de un framework más extenso como Django, y era necesario usar Python para utilizar el modelo

exportado de forma simple. El API tiene solo dos endpoints: la raíz (GET /) que es accesible desde un navegador y le envía al usuario una página web .HTML, y el segundo (POST /comment/predict) que es el que recibe una cadena de texto y envía un objeto con la respuesta de la predicción del modelo.

Implementación de la aplicación y tecnologías usadas

La aplicación web es simple. Se utilizo el templating engine Jinja2 para mostrar un archivo .HTML, el cual es rellenado con meta data del backend. La página es una página estática, utilizando solo HTML, CSS y JS natural (salvo que además se importó la librería Axios para hacer solicitudes asíncronas). Esto se hizo para reducir el tamaño de la página, y para evitar añadir funcionalidad innecesaria. La página recibe el texto, y al enviarlo se hace una solicitud al backend con el texto. Al recibir la solicitud, se muestra el resultado sobre la página (sin tener que cargar otra página) usando JS y CSS. La aplicación es stateless, y no requiere del uso de bases de datos ni persistencia.

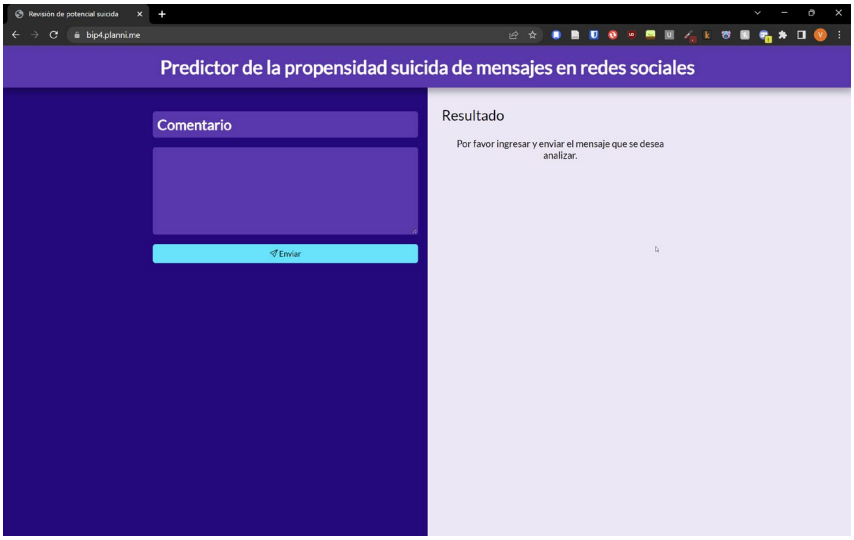
La aplicación fue desplegada usando Docker, con acciones de GitHub para implementar los procesos de CI/CD. El contenedor se ejecuta en un servidor personal que le pertenece a Diego, y se expone al público usando Nginx para gestionar el dominio público y HTTPS. Se encuentra desplegada la página en https://bip4.planni.me.

Desarrollo de la aplicación y justificación

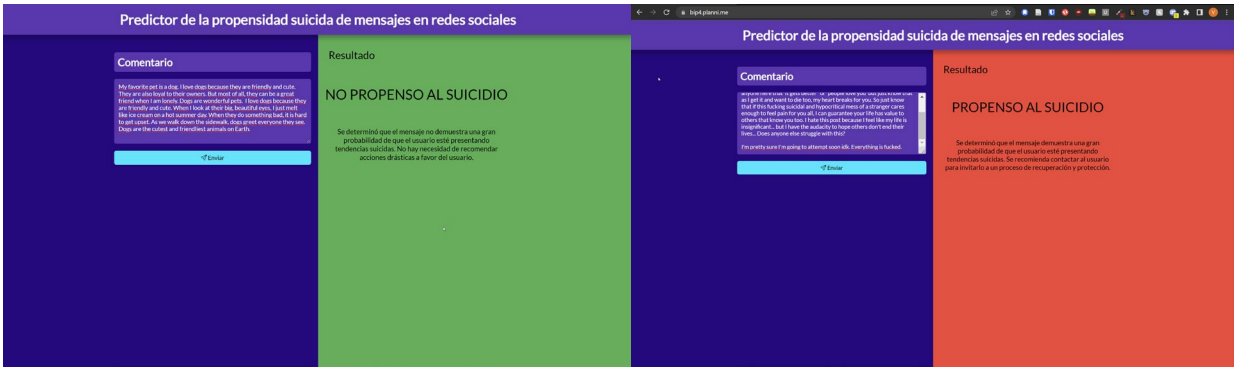
Ahora, el rol del usuario de esta aplicación seria principalmente de un agente de apoyo en algún tipo de línea de apoyo de salud mental o algún entorno que requiera moderación. Como este tipo de usuario esta interactuando con grandes cantidades de información diariamente usar una aplicación con estas características le permitirá analizar con mayor rapidez una mayor cantidad de textos y con esto hacer una evaluación preliminar sobre un texto y el estado mental de su autor. Aunque se intentó asegurar la menor cantidad de falsos negativos en el modelo el papel principal del usuario es el de vetar las decisiones del modelo y de pronto traer una opinión profesional desde el punto de vista de un experto en salud mental o alguien mas familiarizado con el tema y señales mas sutiles que indiquen tendencias suicidas. Una línea de apoyo, servicio de psicología, servicio de moderación para redes sociales serian algunos de los entornos de negocio donde esta aplicación servirá de mayor utilidad para sus usuarios. Como es una herramienta de apoyo a la decisión esta agilizara el proceso de lectura y ayudara a reducir el tiempo que un usuario se demore en tomar una decisión sobre el estado de salud mental del autor de un texto. La facilidad de acceso de esta aplicación es también un elemento importante, ya que hacerla de fácil acceso por medio del internet aumenta su disponibilidad y facilidad de uso.

Resultados

La página https://bip4.planni.me tiene la siguiente apariencia para tanto móvil como desktop.



En esta página se tiene un campo de texto donde se puede ingresar el texto que se desea analizar. Como el set de datos dado para entrenamiento y pruebas estaba originalmente en ingles el modelo solo aplica para párrafos y oraciones en inglés.



Cuando un usuario realiza ingresa un texto y oprime enviar se realiza un request al api para que el modelo procese el texto, una vez esa respuesta es retornada esta se refleja en la página indicando si el autor del texto es propenso al suicidio o no. De esta forma sirviendo como una herramienta de apoyo fácil de usar para hacer el análisis de texto y determinar la propensidad del autor de cometer un intento de suicidio.

En términos de performance entre más largo es el contenido del texto a procesar más tiempo se demora en obtener una respuesta al llamado de la API. Por lo tanto, a pequeña escala este es un servicio que es útil si una persona desea hacer análisis rápido de los datos. Sin embargo, si el objetivo fuese el de cargar una gran cantidad de datos y realizar predicciones en batch sobre sería ideal hacer un despliegue en un clúster de máquinas para lograr aumentar la velocidad de procesamiento, hacer esto no

sería muy difícil si se quisiese usar un servicio cloud como Google Kubernetes Engine, con el cual podríamos generar un clúster con un montón de nodos hijos a base del archivo de Docker que ya tenemos creado para si procesar las requests al api del modelo. Sin embargo, de la forma que se envisiona esta herramienta es como un soporte a un agente humano, por lo tanto, este tipo de interfaz es adecuada y amigable y da respuestas claras sobre la predicción del modelo.

Por lo tanto, como el objetivo del proyecto es crear una herramienta para el apoyo a la clasificación de textos y se presume que se tendrá a un actor humano revisando e interesando la información de los textos se puede determinar que esta herramienta es útil y versátil. Es fácilmente accesible por medio del internet, es flexible en el tipo de modelos que puede usar ya que solo necesita el .pkl para poder implementar un nuevo modelo, hace el procesamiento de una forma organizada y reusable para que cualquier algoritmo de NLP lo pueda usar y da amplias posibilidades para expandir sobre la aplicación a futuro ya sea para procesamiento en batch o para soportar una aplicación a mucha mayor escala.

Video

En el siguiente link se puede consultar el video donde se explica la funcionalidad de la aplicación https://youtu.be/KGjan4ctS1g . Este simula la interacción del usuario final con la aplicación y describir en que tipo de contextos se puede usar y como puede traer valor a moderadores, personal de líneas de salud mental o profesionales de salud mental

Trabajo en equipo

Los roles fueron distribuidos de la siguiente manera. Verónica Escobar fue líder de proyecto e ingeniería de software responsable de diseño de la aplicación, Diego Granada fue ingeniero de software responsable de desarrollar la aplicación final y Andres Felipe Arias Russi fue ingeniero de datos. Veronica Escobar fue la responsable de gestionar la organización y coordinación de la Etapa 2, ayudo a adaptar las funciones de procesamiento de datos a un formato más fácil de usar en el proyecto, diseño parcialmente la aplicación y se encargó de realizar el video. Diego Granada fue encargado principal del desarrollo de la aplicación final, él fue el que implemento el diseño completo de la aplicación y adoptar los modelos hechos en la etapa previa de una forma que funcionase en el contexto de la aplicación. Andres Felipe Arias Russi fue el ingeniero de datos y fue quien velo por la calidad del proceso de automatización a la hora de construir el modelo analítico en la aplicación, ayudo a adaptar el modelo SVM que él había creado en la etapa anterior de una forma organizada y limpia para que fuese fácil de implementar, trabajo de mano en mano con diego en la parte de plantear el modelo en el contexto de la aplicación.

Todos dedicaron aproximadamente 4 horas de trabajo individual para esta etapa del proyecto, al igual que en la etapa 1 no se realizaron reuniones por que no se vio la necesidad y todo se manejó por chat o en discusiones cortas fuera de clase. El reto

principal fue el de adaptar el modelo para ser usado dentro del contexto de la aplicación, puesto la forma que estaba estructurado en los notebooks era muy diferente a como debería ser en un programa tradicional que da cara al cliente, por lo tanto, ese aspecto de automatización fue el más difícil de manejar en esta etapa. Esta etapa fue mucho más sencilla en comparación a la primera porque se pudo comenzar el desarrollo de esta en simultaneo con la etapa 1 aunque sin mucha noción de cómo se iba a automatizar el modelo y su procesamiento. Por lo tanto, para cuando fue la hora de implementar el modelo en la aplicación ya todo lo demás estaba hecho por lo tanto nos pudimos enfocar solo en esta tarea, lo cual consideramos bueno. Aunque se menciona en el enunciado la verdad no vimos muchas oportunidades de enriquecer el modelo analítico, sonaba interesante la idea de integrarlo con otras fuentes de datos, pero debido a limitaciones de tiempo esto no era factible y preferimos priorizar el funcionamiento esencial.

Datos, Resultados y Repositorio

La página se puede acceder con el siguiente enlace: https://bip4.planni.me

El repositorio que contiene todo el código y la wiki está en la siguiente página: https://github.com/vescobars/BI_Proyecto1