Proyecto #1.2

Apoyo al diagnóstico de pacientes: Identificación del problema del paciente a partir de una descripción general dada por un médico

Universidad de Los Andes

ISIS 3425 – Sistemas Empresariales

Integrantes:

Nicolás Segura Castro – Sección 2

1. Introducción

Este proyecto cuenta con el objetivo de automatizar un proceso de clasificación de textos con el fin de construir un modelo que funcione para ayudar al personal médico a clasificar textos médicos en 5 categorías distintas entre lasque están neoplasmas, enfermedades del sistema digestivo, enfermedades del sistema nervioso, enfermedades cardiovasculares y condiciones patológicas generales. Para tal fin, fue desarrollada una aplicación web que pudiese implementar el modelo anteriormente construido en la etapa 1 con el fin de servir como una herramienta de ayuda para el diagnóstico de los pacientes. En este informe se podrá ver qué pasos fueron realizados para el desarrollo de la aplicación.

2. Proceso de automatización

Para empezar con el proceso de automatización, el primer y uno de los pasos más importantes, fue el de exportar y adecuar los pipelines proporcionados por la librería joblib con el fin de asegurarse de que todo el trabajo realizado previamente funciona de manera correcta. Para dicha adecuación se prepararon los datos mediante su preprocesamiento y perfilamiento, por lo que la primera tarea consistió en eliminar el ruido de la entrada convirtiendo todos los caracteres a minúsculas, eliminando los caracteres extraños que no pertenecieran al inglés, eliminación de los signos de puntuación y exclamación, sacando las palabras de poca relevancia como las preposiciones, los artículos y los pronombres, quitando los números y eliminando los espacios sobrantes también.

Una vez limpiado el ruido, se realizó la tokenización de las palabras de los textos separándolas en tokens únicos y creando una lista con todas las palabras relevantes de cada uno de los textos. Luego, se continúo con la normalización de los datos a partir de la lematización de los datos ingresados por los usuarios a partir de la eliminación de prefijos y sufijos y la lematización de los verbos. Para la preparación de las palabras se utilizó el algoritmo de lematización de Porter2 por sus tiempos y resultados vistos.

Para continuar con la adecuación del pipeline se decidió implementar un único modelo para incluir en la aplicación y facilitar el trabajo. Si bien es cierto que en la primera versión del proyecto se implementar los algoritmos de regresión logística, MLP Clasiffier y clasificación por medio de árboles, se decidió utilizar el pipeline asociado con el modelo de regresión logística por sus buenos resultados asociados con la clasificación de enfermedades del sistema digestivo y aquellas asociadas con el sistema cardiovascular. Es importante tener en cuenta que uno de los factores más importantes para el proyecto es que las clasificaciones sean lo más precisas posibles ya que se tratan de clasificaciones de diagnósticos de pacientes.

Una de las tareas más complicadas fue la de la vectorización de los datos, que se realizó por medio de modelos neuronales que fueron entrenadas a partir de modelos que previamente ya lo estaban sobre bases de datos relacionadas con el lenguaje médico. Además de lo anterior, fue definida una clase para cargar un modelo entrenado de la librería BioSentVec para embeber

los textos en vectores que pudiesen ser entendidos durante el proceso de automatización. Es importante recalcar que también se utilizaron modelos de TensorFlow que requirieron un arduo trabajo para integrarlos en la aplicación. Por último, se procedió a almacenar a almacenar el modelo pre-entrenado y a integrarlo con el API de FastAPI para que pudiera ser utilizado posteriormente por la aplicación desarrollada en Angular.

3. Desarrollo del API

Para desarrollar el back de la apl importante recalcar que también se utilizaron modelos de TensorFlow que requirieron un arduo trabajo para integrarlos en la aplicación. Por último, se procedió a almacenar a almacenar el modelo pre-entrenado y a integrarlo con el API de FastAPI para que pudiera ser utilizado posteriormente por la aplicación desarrollada en Angular. Para el despliegue del API fueron creados varios módulos entre los que están clases.py, que posee las dependencias para la etapa de preprocesamiento como los métodos que eliminan los caracteres innecesarios como las tildes y convierte todas las palabras en minúsculas. Además, fue creado el archivo dataModel.py para corresponder a los datos mapeados con los datos recibidos desde el front. Este proceso resultó de gran importancia ya que la clase cuenta con las mismas características que el dataset con el que fue entrenado el modelo.

Una vez creados los módulos se procedió con el despliegue, el cual puede accederse a través del endpoint dispuesto /predict en la que la aplicación recibe un DataModel. Luego de recibir la petición, se procede a crear un dataframe con los datos y el pipeline es cargado luego de agregar el modelo de TensorFlow. Una vez realizados los pasos, se predicen las probabilidades pertenecientes a cada categoría y se pueden visualizar en la página web.

4. Roles

Para el desarrollo de la aplicación fue únicamente considerado un rol con el fin de garantizar un funcionamiento óptimo y sencillo de la aplicación. Este rol es el del médico principal, que es el que debe ingresar los textos a la aplicación para poder conocer las probabilidades asociadas al diagnóstico. Sin embargo, se debe tener en cuenta que resultaría extremadamente interesante que exista otro rol para un médico asistente que se encargue de alimentar al modelo con datos históricos de los pacientes con el fin de que haya más información útil para alimentar el modelo y que, de esta manera, tenga eventualmente con el tiempo mayores capacidades para predecir las categorías con mayor exactitud.

5. Distribución de trabajo

Para este proyecto, únicamente una persona realizó todas las tareas, por lo que fue extremadamente complicado el poder cumplir con las metas planteadas a cabalidad. De esta manera, fue Nicolás Segura Castro quien se encargó del desarrollo de la aplicación tanto en

front como en back, de la preparación del pipeline y el entrenamiento de los modelos. Es por lo anterior que los 100 puntos de trabajo en equipo no se repartieron.