**Proyecto 1 Entrega 2**

F. Bedoya, C. Salinas, N. Orjuela

**Grupo #20**

**Sección #1**

**Inteligencia de Negocios**

**Departamento de Ingeniería de Sistemas y Computación**

**Universidad de Los Andes**

**Bogotá, Colombia**

1. **Proceso de automatización del proceso de preparación de datos, construcción del modelo, persistencia del modelo y acceso por medio de API**

A

Para el desarrollo del API se utilizó el framework FastAPI de Python, el mismo utilizado en el laboratorio 4. En total hay 6 endpoints, 2 para cada modelo desarrollado en la primera entrega del proyecto; estos 2 endpoints son “/predict” y “/getrsquared”, el primero recibe un diagnostico y el modelo devuelve la clasificación de este (si es elegible o no) y el segundo recibe un conjunto de diagnósticos y la clasificación de cada uno, y se devuelve el R2 del modelo.

1. **Desarrollo de la aplicación y justificación**

Para el desarrollo de la aplicación se tuvo en cuenta un usuario especifico y todas las decisiones de diseño se orientaron hacia este rol. Después de investigar los procesos de selección reales en el negocio se observo que en Colombia normalmente son los médicos los que toman las decisiones y analizan quienes pueden pertenecer o no a un estudio. Muchas veces son médicos en rural o de poca experiencia en su campo medico como en conocimientos técnicos. Así mismo se considero que en Estados Unidos, importante porque los inputs se consideraron en inglés, muchas veces los que toman este tipo de decisiones de pertenencia son administrativos con pocos conocimientos médicos y nivel básico en ofimática.

Es de suma importancia considerar el objetivo de la aplicación dentro de la organización. Si bien no va a reemplazar la intuición y juzgamiento de un ser humano, puede servir para obviar aquella capa de conocimientos técnicos en oncología y especializaciones que puede no tener un médico general recién graduado o un administrativo.

Con esto en mente se diseñó una aplicación Web muy amigable para cualquier usuario en donde se puede tener registro de todos los pacientes que fueron evaluados para la prueba médica. Se puede reevaluar y revisar aquellos que ya obtuvieron una predicción de los modelos, pero también generar nuevas predicciones desde cero. Como funcionalidad adicional se integraron los tres modelos entrenados, uno rápido, pero menos preciso, uno intermedio y uno lento, pero mas preciso. La aplicación en React accede al FastApi y utiliza los joblibs mencionados anteriormente.

1. **Resultados**

A

El video se encuentra en el Padlet destinado

1. **Trabajo en equipo**

* **Líder de proyecto:** Nicolás Orjuela
* **Ingeniero de datos:** Felipe Bedoya
* **Ingeniero de software responsable del diseño de la aplicación y resultados:** Camilo Salinas
* **Ingeniero de software responsable de desarrollar la aplicación final:** Camilo Salinas

1. Retos:

Empezar a desarrollar en una nueva tecnología siempre es un reto y nos enfrentamos a varios problemas que nos retrasaron en el flujo de trabajo que teníamos planteado, especialmente en FastAPI. Generar pruebas con los datos de entrenamiento y leer la documentación ayudaron mucho a solucionar los pequeños bugs que encontramos en el camino.

A la hora de integrar el FastAPI y la aplicación en React también hubo algunos problemas cuando se hacían peticiones. En la última reunión de consolidación que tuvimos nos unimos para resolverlo corriendo todo el stack de manera local y recreando los errores que teníamos.

1. Tareas realizadas:

Creación de la aplicación en React – Camilo Salinas

Creación del FastAPI – Nicolas Orjuela

Mantenimiento y exportación del pipeline – Felipe Bedoya

1. Repartición de puntos: