Prueba técnica Gestor SR con énfasis en IA.

Practica ciencia de datos y Machine learning:

Summa-sci requiere construir un servicio (Backend) para una aplicación de experiencia del colaborador, esta aplicación lo que hace es recibir en la interfaz de usuario (Frontend) unos parámetros de una lista desplegable ingresados por el usuario y con esto un valor en números que corresponde al pronóstico de la demanda de las compras de la compañía Cementos Argos, asociado a esos parámetros.

La información es enviada desde la interfaz hacía el servicio, este la recibe en formato json y utiliza estos datos para generar una clasificación (Alpha, Beta) que son códigos que le indican al analista que tipo de compras de materiales debe hacer, y en que cantidades, finalmente el servicio debe responderle a la interfaz de usuario con un json en el que se especifica cual es el código.

Hay 3 archivo útiles para realizar este proyecto:

`dataset_demand_acumulate.csv`

Este archivo contiene la información de la demanda entre el 2017-01 hasta el 2022-04 siendo (año-mes).

`dataset_alpha_betha.csv`

Este archivo contiene la información de todas las variables involucradas para realizar la clasificación de si un registro es Alpha o Betha, este cuenta con más de 7000 registros.

`to_predict.csv`

Este archivo es lo que buscamos predecir, cuenta con 3 registros los cuales ya tienen toda la información completa, excepto la demanda y la clase.

Para este proyecto usted deberá:

1. Deberá crear un modelo que pueda pronosticar la demanda de 2022-05, 06 y 07, de Cementos Argos. El código de esto deberá hacerlo en un Jupyter notebook, el output debe ser un archivo con todos los datos incluyendo los pronósticos, además de un gráfico donde se pueda apreciar cual fue su data de entrenamiento, de validación y pronóstico, dentro de este gráfico se debe apreciar las métricas con las que decidió que su modelo estaría dando un resultado confiable (Puede usar las que considere).

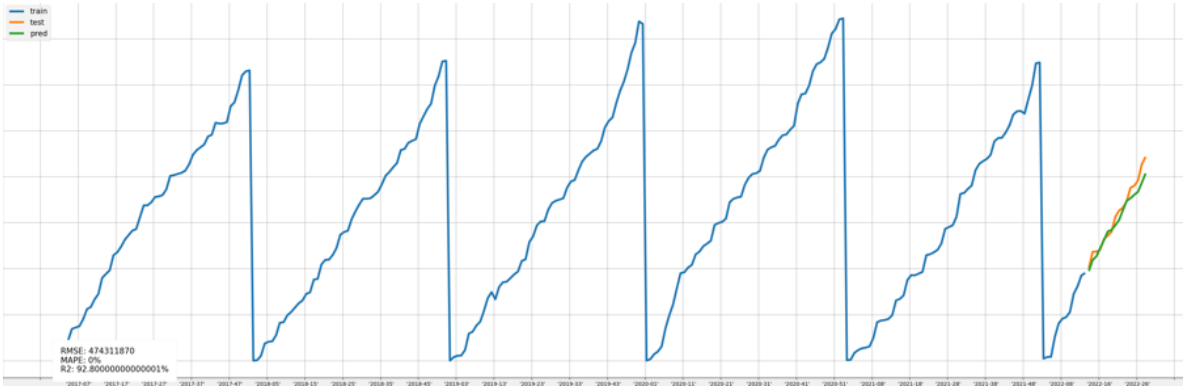

Ilustración 1. Ejemplo visualización

2. Deberá entrenar un modelo de clasificación para generar las respuestas que hará el servicio, el código deberá hacerlo en un Jupyter notebook, el output de esta ejecución será un archivo con el modelo entrenado y un txt con las métricas generadas por este, también indicaciones de que modelo usó, cantidad de datos de entrenamiento, testeo, etc. Es libre de hacer reducción de dimensionalidad, escalado, aumento de data, balanceo de data o cualquier técnica que mejore su pronóstico. (Tenga en cuenta que si redujo la cantidad de variables predictoras para el momento de hacer el pronóstico debe hacerlo también a esos datos)

3. Deberá crear un proyecto en el cual exponga un api (Puede usar flask, FastApi, django, o cualquier librería para exponer apis) y sea ejecutable desde un archivo main.py, este proyecto deberá consumir el modelo que creó en el punto 2, recibir el json de una solicitud post, realizar la clasificación y entregar la respuesta. El entregable será el repositorio del proyecto y unas instrucciones para ejecutarlo, con archivo readme y requirements. Tanto el json del post como el del output usted puede organizarlos como quiera, solo necesita que tenga las variables que el modelo usa.

4. Complete el archivo “to_predict.csv” con la información de la demanda que pronosticó en el punto 1. Y realice las 3 peticiones al servicio que diseñó, finalmente entregue un archivo con los resultados que generó su servicio con el modelo predictivo.

5. **(Opcional) De ser posible, cree un contenedor en Docker con la aplicación, así puede ser ejecutada y desplegada en ambiente productivo de manera más sencilla, deberá entregar el Docker file (Docker compose), para que nuestro equipo haga el build y ejecute este contenedor. (Opcional)**

6. Los entregables de este proyecto deben subirse a su repositorio de github, trate de hacer buen manejo de las ramas, commits, pr, etc, y también buen manejo de las carpetas dentro del repositorio, el archivo de respuestas teóricas debe ser subido en formato pdf con el nombre “teoria_.pdf”

Practica GEN AI:

La compañía requiere construir un agente de chat sencillo para responder ciertas solicitudes de gestión humana, este debe tomar la necesidad del usuario y responder según lo que se requiera, para esto le piden a usted que haga una prueba de concepto de como funcionaría el agente y le entregan un documento de Excel con información de las cesantías causadas de los usuarios llamado.

`Cesantías_causas.xlsx`

1. Construye una prueba de concepto donde un usuario pueda agregar una pregunta (Puede ser directamente en el código o en consola) y un modelo de llm decida que hacer con esta, las posibilidades son.
    a. Responder la pregunta del usuario con el conocimiento base del agente, tratar de no desviarse demasiado y tener el contexto de la compañía al responder.
    b. Revisar el archivo de censarias causadas y responder la pregunta del usuario con la información del archivo.

La idea es que uses cualquier modelo de lenguaje, o si no tienes como, dejar expresado su uso y explicarlo. Para las acciones que debe tomar el agente usa tecnologías MCPs y maneja la lógica interna de los modelos de lenguaje.

El resultado de esta parte de la prueba si tienes como ejecutarla, puede ser un txt con 4 preguntas, sus respectivos procesos de pensamiento y resultados.

Sino tienes como ejecutar los modelos de lenguaje, el resultado solo se deja expresado y deberá ser explicado en la sustentación.

2. Entrega un Diagrama de la arquitectura y componentes que vas a utilizar para atender este problema, y si tuvieras acceso a todas las herramientas de la nube y cualquier modelo de lenguaje cual usarías y porqué.