1. Clasificación de Correos en el Área de Compras

Metodología y Plan de Proyecto

1. Entendimiento de la necesidad con el área de compras
    - El primer paso del proyecto es entender mejor la necesidad del área de compras, que proceso desean mejorar y por qué. Además, se hace importante conocer que resultado espera el área y definir si la implementación de esta solución va ligada a mejorar una métrica operacional en específico. Al tener más contexto, la solución a desarrollar será mucho más exacta respecto a las necesidades de GA.
2. Recolección de Datos:
    - Recolectar los correos históricos de las diferentes categorías.
    - Aquí se parte del supuesto de que los correos actuales estén bien etiquetados. En caso de que este no sea el caso, se hace necesario un paso intermedio de clasificación de correos asignándole una de las categorías establecidas por medio de algoritmos de procesamiento de lenguaje natural (PLN) dedicados a la clasificación de texto.
3. Exploración y Limpieza de Datos:
    - Realizar un análisis exploratorio de los datos para entender la distribución y las características de los correos.
    - Limpieza y preprocesamiento de datos: eliminación de duplicados, normalización del cuerpo y encabezado de los correos, lo cual implica la eliminación de caracteres especiales, dobles espacios, pasar el texto a minúsculas, etc. El uso de librerías de expresiones regulares puede facilitar este trabajo (librería *re*, en Python).
    - Además, se plantea establecer diferentes versiones de los datos (una vez que han sido limpiados) donde lleven procesos de tokenización, lematización, y eliminación de stop words. Esto facilitará la iteración de todas estas versiones en diferentes algoritmos de clasificación de PLN.
4. División de Datos:
    - Dividir los datos en conjuntos de entrenamiento (70%), validación (15%) y prueba (15%).
5. Selección y Entrenamiento de Modelos:
    - Utilizar modelos de procesamiento de lenguaje natural y clasificación de texto como Naive Bayes, Support Vector Machines (SVM), Random Forest y Transformers (por ejemplo, BERT). Además, por cada tipo de modelo se pueden iterar las diferentes versiones de los datos.
    - Realizar tuning de hiperparámetros utilizando validación cruzada.

6. Evaluación de Modelos:
    - Evaluar los modelos utilizando métricas como precisión, recall, F1-score, y matriz de confusión.
    - Seleccionar el modelo que tenga mejor desempeño en el conjunto de validación.
7. Despliegue del Modelo:
    - Implementar el modelo seleccionado en un entorno de producción.
    - Integrar el modelo con el sistema de correo electrónico para la clasificación automática.
8. Monitoreo y Mantenimiento:
    - Configurar un sistema de monitoreo para evaluar continuamente el desempeño del modelo.
    - Actualizar el modelo periódicamente con nuevos datos y reevaluar su rendimiento.
    - Si en principio se definió una métrica operacional por parte del área de compras, se hace tambien necesario evaluar y monitorear dicha métrica una vez que el modelo ha sido desplegado en el ambiente productivo.

Arquitectura del Proyecto

1. Ingesta de datos:

   Utilizar pipelines para recolectar y preprocesar los correos electrónicos.
2. Preprocesamiento de datos:
    - Scripts en Python utilizando librerías como NLTK, SpaCy, y Scikit-learn para limpieza y preprocesamiento.
3. Almacenamiento:
    - Base de datos SQL o NoSQL para almacenar los correos preprocesados y anotados.
4. Modelo de clasificación:
    - Modelos entrenados utilizando frameworks como Scikit-learn, TensorFlow, o PyTorch.
5. Despliegue:
    - Despliegue del modelo mediante una API REST utilizando Flask o FastAPI para permitir la integración con el sistema de correo.
    - Utilizar contenedores Docker para portabilidad y fácil despliegue.

6. Monitoreo:
    - Herramientas como Prometheus y Grafana para el monitoreo del rendimiento del modelo y la infraestructura.

Recursos humanos necesarios:

- Arquitecto de datos, el cual será fundamental para garantizar una adecuada infraestructura de datos y pipelines.
- Científico de datos: encargado del proceso de EDA, construcción del modelo de clasificación de correos.
- Ingeniero de ML: el cual se encargará de probar el correcto funcionamiento del modelo, así como de revisar la correcta infraestructura del mismo.
- Ingeniero de software: facilitara la integración del modelo de clasificación con los sistemas de correo del área de compras.
- Área de compras: su presencia es fundamental en todo el proceso de construcción, hasta la etapa de monitoreo.

Recursos tecnológicos necesarios:

- Computador tipo work-station, con amplias capacidades de computo: RAM, GPU, CPU, etc.
- Acceso a herramientas en nube, con el fin de tener a disposición VM, espacios de orquestación de datos, etc.
- Python, Docker, servicio en la nube, VM.

2. Identificación y Corrección del Drift en un Modelo de Regresión

Sí, es posible que el modelo esté sufriendo algún tipo de drift. Los cambios en los datos o en las relaciones subyacentes pueden hacer que las predicciones del modelo se vuelvan menos precisas con el tiempo.

Validación de drift

- Monitorización de Predicciones y Realidad:
    - Comparar las predicciones del modelo con los valores reales a lo largo del tiempo.
    - Identificar desviaciones significativas utilizando métricas como el error absoluto medio (MAE), el error cuadrático medio (MSE) y la raíz del error cuadrático medio (RMSE).
- Análisis de Distribución:
    - Analizar si ha habido cambios en la distribución de las características de entrada (features) y las etiquetas (targets).
    - Utilizar técnicas como KS-test o AD-test para identificar cambios en las distribuciones.

- Validación Continua:
    - Dividir los datos recientes en intervalos de tiempo y evaluar el rendimiento del modelo en cada intervalo.
    - Realizar pruebas estadísticas para detectar drift (por ejemplo, Drift Detection Methods como Page-Hinkley).

Corrección del drift

Si se confirma que el modelo está sufriendo drift, estas son algunas acciones que se pueden implementar:

- Reentrenar el modelo con datos más recientes que reflejen mejor las condiciones actuales. Esto ayudará a que el modelo aprenda las nuevas relaciones o patrones que han emergido.
- Implementar un pipeline de actualización continua donde el modelo se entrene periódicamente con datos nuevos.
- Revisar y ajustar los hiperparámetros del modelo para asegurar que sigue optimizado para las condiciones actuales.
- Implementar un sistema de monitoreo continuo que detecte automáticamente cuando caiga el desempeño del modelo y envíe alertas.
- Revisar las características utilizadas en el modelo. Es posible que se necesite crear nuevas características o transformar las existentes para capturar mejor las nuevas relaciones en los datos.

3. Asegurar la Pertinencia de Respuestas del Chatbot GPT-3.5

Estrategias para controlar respuestas

- Utilizar prompt engineering para proporcionar instrucciones claras al modelo GPT-3.5 sobre el tipo de información que debe proporcionar.
- Implementar una lista de temas permitidos y utilizar filtros para asegurar que las respuestas generadas se mantengan dentro de estos temas.
- Fine-tuning del modelo GPT-3.5 utilizando un conjunto de datos específico que solo incluya ejemplos de respuestas relevantes.
- Implementar un módulo de validación que revise las respuestas generadas y asegure su pertinencia antes de enviarlas al usuario.
- Utilizar técnicas de NLP para validar la coherencia y relevancia de las respuestas.
- Recoger feedback de los usuarios y utilizarlo para mejorar continuamente el modelo y su capacidad de mantenerse en el tema.