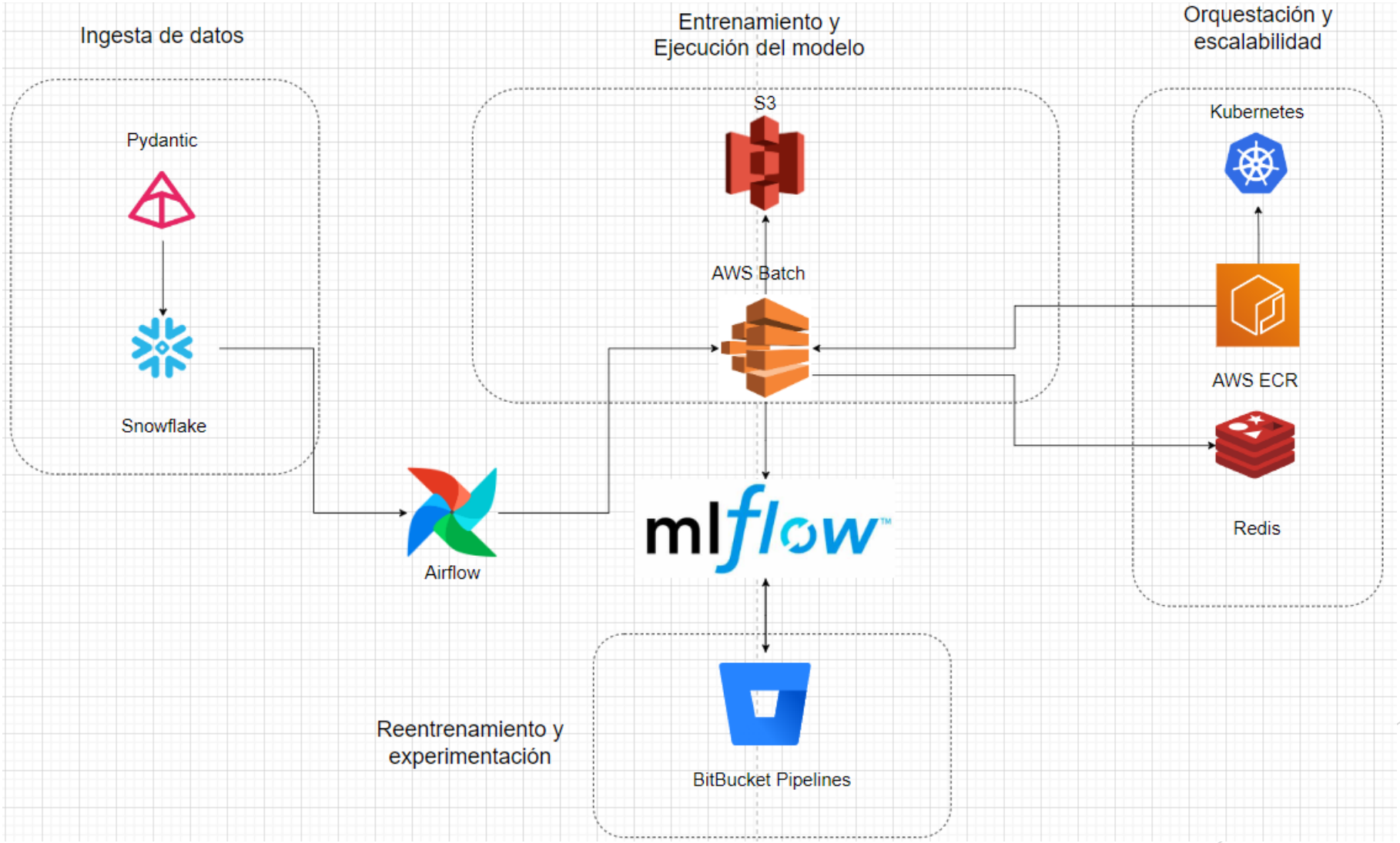
Parte 1- Rappi Technical Challenge

Alberto Mendoza

Retos de despliegue

- Predecir compras de los usuarios en los próximos 7 días
- Se desplegará en 9 países y a 50 millones de usuarios
- 5 modelos por país 45 en total

Diagrama de infraestructura



Explicación de componentes

- AWS ECR almacena las imágenes de Docker para las versiones del modelo
- Kubernetes manejo del despliegue de las imágenes en diferentes instancias de EC2
- AWS Batch para manejar tareas de predicción por eventos diarios
- Pydantic para validaciones de los datos entrantes con esquemas predifinidos
- Snowflake para la persistencia de la información cruda, transformada y las predicciones, Snowflake streams para monitorear

Explicación de componentes

- Airflow para orquestar las predicciones de AWS Batch cuando termina el pipeline de snowflake
- MLFLOW para manejar modelos, versiones y monitorear rendimientos
- Bitbucket pipelines para la automatización CI/CD, corriendo tests sobre los datos y automatizando el despliegue de los algoritmos
- Redis para almacenar las predicciones que son solicitadas constantemente
- AWS S3 para el almacenamiento de datasets de entrenamiento y artefactos del modelo

Estrategias para reentrenamiento y seguimiento de métricas

- MLFLOW para el monitoreo de versiones de los modelos y experimentos
- El CI/CD automático con bitbucket pipelines
- Probar el despliegue de los modelos con segmentos más pequeños del tráfico para verficar el comportamiento
- Seguimiento del rendimiento del modelo y agendar reentrenamientos periódicos o basado en los resultados
- Usar AWS S3 para almacenar datasets de entrenamiento y artefactos del modelo

Asuntos de calidad de datos y data drift

- Implementar logging en airflow para detectar anomalías o datos faltantes
- Usar Pydantic para chequeos y validaciones sobre los esquemas de los datos
- Evaluar el desarrollo de métricas de data drift personalizadas coherentes con la tarea de clasificación y los datos
- Monitoreo de las métricas de rendimiento constante con MLFOW
- Evaluar despliegues con aws Cloudwatch para generar alertas de calidad de datos y buen funcionamiento del sistema

Consideraciones escalabilidad

- Escalamiento automático de Kubernetes para añadir instancias dependiendo de la carga requerida
- S3 para el almacenamiento de logs, resultados intermedios y artefactos del modelo
- AWS Batch procesará las predicciones en paralelo con las instancias EC2
- Redis para requests de predicciones de alta frecuencia para reducir latencia

PARTE 2 PRUEBA

Link de aplicación

https://takenorderclassifierrappi-nrsvevhywwtkabagm9ykqj.streamlit.app/

Link de repositorio

https://github.com/Alberstation/taken_order_task/tree/main