Cloud, MLops, productivización de modelos – Caso Práctico

1 - Describir la forma de productivizar cada modelo. ¿Qué tipo de herramienta se debería elegir para desarrollar cada modelo y por qué se ha elegido? ¿Cuál sería la forma de acceder a los modelos por parte de los usuarios? ¿Cada cuánto tendrían que reentrenarse?

Para productivizar los modelos de ML y permitir que los usuarios accedan a ellos, se puede considerar el siguiente enfoque para cada modelo en el escenario dado:

- **Conversión de base de datos a CSV:**
  - Para las funciones sin servidor que copian la información de la base de datos a archivos CSV, se pueden utilizar servicios en la nube proporcionados por el proveedor de servicios en la nube elegido, en este caso Microsoft Azure. Azure Functions, junto con Azure Blob Storage o Azure Data Lake Storage, pueden utilizarse para desarrollar e implementar las funciones sin servidor en un sistema de almacenamiento distribuido. Azure Functions proporciona una arquitectura basada en eventos que puede activarse diariamente para realizar la conversión de base de datos a CSV.

- **Modelo de predicción de ventas:**
  - Para desarrollar el algoritmo de ML para predecir las ventas futuras existen varias opciones según la complejidad y los requisitos específicos del modelo:
    Se pueden utilizar bibliotecas y marcos de trabajo basados en Python, como scikit-learn, TensorFlow o PyTorch. Estos *frameworks* proporcionan una amplia gama de herramientas y algoritmos predefinidos para tareas de regresión.
    Azure Machine Learning puede utilizarse para el desarrollo, experimentación y entrenamiento del modelo. Azure ML proporciona un entorno gestionado con integración incorporada para marcos de ML populares, capacidades de aprendizaje automático automatizado e infraestructura escalable para el entrenamiento.
    El modelo se puede exponer como un servicio web utilizando las capacidades de implementación de Azure ML. Azure ML te permite implementar modelos como API o servicios contenerizados, que los usuarios pueden acceder a través de llamadas a la API.
    El modelo de predicción de ventas debe reentrenarse periódicamente para incorporar nuevos datos y adaptarse a los cambios en los patrones. La frecuencia de reentrenamiento puede variar según la estabilidad de los datos subyacentes y los requisitos comerciales. Puede oscilar entre ciclos de reentrenamiento semanales a mensuales, dependiendo de la frecuencia con la que ocurran cambios significativos en los datos o en el entorno comercial.

- **Modelo de predicción de compras telefónicas:**
  - Similar al modelo de predicción de ventas, se pueden utilizar bibliotecas basadas en Python o Azure ML para desarrollar y entrenar el algoritmo de ML para predecir clientes con altas probabilidades de comprar productos por teléfono. La elección de las herramientas dependerá de los requisitos específicos y la complejidad del modelo. También se pueden aprovechar las capacidades de Azure ML para experimentación, entrenamiento e implementación de este modelo.

La frecuencia de reentrenamiento para el modelo de predicción de compras telefónicas puede ser similar a la del modelo de predicción de ventas, dependiendo de la tasa de cambio en el comportamiento de los clientes y la disponibilidad de datos.

- **Algoritmos de análisis empresarial:**
  - Para los algoritmos que se entrenan y ejecutan en puntos específicos para determinar aspectos determinados del negocio, la elección de las herramientas depende de la naturaleza y complejidad de los algoritmos. Algunos factores a considerar son el tipo de análisis requerido, el tamaño y complejidad de los datos y los algoritmos específicos que se utilizan.
    Se pueden utilizar bibliotecas basadas en Python como pandas, NumPy y scikit-learn para manipulación de datos, análisis exploratorio y ejecución de diversos algoritmos estadísticos. Además, se pueden emplear servicios en la nube como Azure Databricks o Azure Synapse Analytics para el procesamiento escalable de datos, la computación distribuida y la ejecución de algoritmos de análisis complejos en conjuntos de datos grandes.
    La frecuencia de reentrenamiento para estos algoritmos de análisis empresarial depende de los algoritmos específicos y la frecuencia con la que se disponga de nuevos datos. Puede variar desde ejecuciones ad hoc cuando llegan nuevos datos hasta reentrenamientos programados en función de las necesidades comerciales y la dinámica de los datos.

En cuanto al acceso de los usuarios a los modelos, el enfoque preferido es exponer los modelos como APIs o servicios web. Los usuarios pueden interactuar con los modelos mediante llamadas a la API, ya sea a través de una interfaz de usuario o integrando los puntos finales de la API en otras aplicaciones o sistemas.

2 - Describir el flujo de trabajo MLOps de los modelos. ¿Sería necesario en todos los modelos?

El flujo de trabajo de MLOps (Machine Learning Operations) se centra en la operacionalización y gestión de modelos de ML a lo largo de su ciclo de vida. Si bien el alcance de la implementación de MLOps puede variar según los requisitos específicos y la complejidad de cada modelo, generalmente se recomienda incorporar prácticas de MLOps para todos los modelos de ML con el fin de garantizar eficiencia, escalabilidad y mantenibilidad. Veamos el flujo de trabajo de MLOps para los modelos en el caso planteado:

- **Conversión de base de datos a CSV:**
  - Dado que esta tarea implica funciones sin servidor que se ejecutan diariamente para copiar la información de la base de datos a archivos CSV, es posible que el flujo de trabajo de MLOps no sea tan crítico para este proceso específico. Sin embargo, sigue siendo beneficioso aplicar control de versiones al código de la función sin servidor y asegurarse de contar con un monitoreo y registro adecuados para realizar un seguimiento de su ejecución y rendimiento.

- **Modelo de predicción de ventas:**
  - Para el modelo de predicción de ventas es muy recomendable incorporar prácticas de MLOps. El flujo de trabajo de MLOps para este modelo generalmente involucraría los siguientes pasos:

        - Recopilación y preparación de datos: Reunir datos relevantes, preprocesarlos y realizar ingeniería de características.
        - Desarrollo y entrenamiento del modelo: Seleccionar algoritmos de ML apropiados, entrenar el modelo utilizando datos históricos de ventas y evaluar su rendimiento.
        - Implementación del modelo: Implementar el modelo entrenado como un servicio o API, lo que permite realizar predicciones.

        - Monitoreo y registro: Implementar soluciones de monitoreo para realizar un seguimiento del rendimiento del modelo, identificar anomalías y detectar cambios en los datos.
        - Integración continua y despliegue continuo (CI/CD): Establecer un pipeline automatizado de CI/CD para agilizar las actualizaciones y despliegues del modelo.
        - Re-entrenamiento y actualizaciones del modelo: Definir *triggers* de reentrenamiento según los requisitos comerciales o los cambios en los datos y actualizar periódicamente el modelo con datos actualizados.

- **Modelo de predicción de compras telefónicas:**
    - Similar al modelo de predicción de ventas, el flujo de trabajo de MLOps es aplicable al modelo de predicción de compras telefónicas. Los pasos involucrados en el flujo de trabajo de MLOps serían similares, incluyendo la recopilación de datos, el desarrollo del modelo, la implementación, el monitoreo y el reentrenamiento. Esto garantiza que el modelo siga siendo preciso y esté actualizado, considerando los cambios en el comportamiento y los patrones de los clientes.

- **Algoritmos de análisis empresarial:**
    - El flujo de trabajo de MLOps para los algoritmos de análisis empresarial puede tener algunas variaciones según los algoritmos específicos y sus requisitos. Si bien los modelos no necesariamente requieren capacidades de predicción en línea, el flujo de trabajo de MLOps aún puede ser valioso en términos de control de versiones, reproducibilidad y monitoreo. Implementar pipelines de datos adecuados, pruebas automatizadas y mantener repositorios de código contribuiría a la eficiencia y confiabilidad general de los algoritmos de análisis.

En resumen, si bien el nivel de implementación de MLOps puede variar para cada modelo, generalmente se recomienda incorporar prácticas de MLOps para garantizar la reproducibilidad, escalabilidad y confiabilidad de los modelos de Machine Learning. Esto permite un desarrollo, implementación, monitoreo y mantenimiento eficientes de los modelos, asegurando que sean precisos y efectivos a lo largo del tiempo.

3 - ¿Qué cambios habría que hacer en los apartados anteriores si la empresa, en vez de 15 trabajadores, tuviese 10000?

Si la empresa tuviera 1000 trabajadores en lugar de 15, habría ciertos cambios y consideraciones que se deberían implementar en las respuestas anteriores. Veamos las modificaciones requeridas para cada aspecto:

- **Estrategia de análisis en la nube:**
    - La estrategia de análisis en la nube seguiría siendo en gran parte la misma en términos de aprovechar las funciones sin servidor y utilizar servicios en la nube como Azure para

la conversión de base de datos a CSV. Sin embargo, se deben tener en cuenta consideraciones para garantizar que la infraestructura pueda manejar el aumento del volumen de datos, los requisitos de cómputo y el acceso concurrente de usuarios. Escalar los recursos en la nube en consecuencia, como utilizar opciones de almacenamiento más grandes e instancias de cómputo ampliadas, sería necesario para respaldar la plantilla más numerosa y sus necesidades de análisis.

- **Desarrollo de modelos de ML y selección de herramientas:**
  - Al considerar los modelos de ML y la selección de herramientas, una plantilla más numerosa introduce la necesidad de mayor eficiencia y escalabilidad. Algunos cambios a considerar son:
    - Aprovechar los frameworks de computación distribuida: Con un conjunto de datos más grande y mayores requisitos computacionales, utilizar marcos de computación distribuida como Apache Spark o Azure Databricks puede ayudar a manejar la escala y los requisitos de procesamiento en paralelo.
    - AutoML y Pipelines automatizados: Implementar herramientas de ML automáticas, como Azure AutoML, puede ayudar a agilizar el proceso de desarrollo del modelo y reducir el esfuerzo manual requerido para la ingeniería de características y la selección de algoritmos.
    - Optimización de modelos: Con una plantilla mucho más numerosa, la empresa podría permitirse asignar más recursos al desarrollo de los modelos de Machine Learning. Podrían desarrollarse varios modelos simultáneamente para la misma tarea, compararlos y elegir el que funcione mejor, así como dedicar más tiempo a la optimización de hiperparámetros.
    - Entrenamiento de modelos escalables: Asegurarse de que la infraestructura de entrenamiento de ML esté diseñada para manejar el conjunto de datos más grande y los mayores requisitos computacionales. Esto puede implicar el uso de técnicas de entrenamiento distribuido o aprovechar los servicios de ML en la nube que ofrecen capacidades de entrenamiento escalables.
- **Acceso y reentrenamiento de modelos:**
  - Con una plantilla más numerosa, se deben considerar aspectos para garantizar un acceso fluido a los modelos de ML y su reentrenamiento:
    - Escalado de API y balanceo de carga: Escalar las APIs de los modelos para manejar el mayor acceso concurrente de usuarios e implementar mecanismos de balanceo de carga para distribuir la carga de trabajo de manera efectiva.
    - Reentrenamiento automatizado: A medida que aumentan el conjunto de datos y las interacciones de los usuarios, es posible que los modelos necesiten un reentrenamiento más frecuente para mantenerse actualizados. Implementar pipelines de reentrenamiento automatizado, desencadenados por condiciones específicas (por ejemplo, llegada de nuevos datos o intervalos de tiempo predefinidos), garantizaría que los modelos se mantengan precisos.

- **Pipeline de DevOps:**
  - Con una plantilla más numerosa, el pipeline de DevOps debe escalar en consecuencia para adaptarse a los mayores requisitos de desarrollo, prueba e implementación. La infraestructura debe diseñarse para manejar cargas de trabajo más grandes y un mayor tráfico. Los procesos de CI/CD deben optimizarse para manejar una base de código más grande y actualizaciones más frecuentes. Además, se necesitarían sistemas de monitoreo y registro más robustos para administrar y rastrear la mayor escala de los sistemas.

En resumen, con una fuerza laboral de 1000 personas, el pipeline de DevOps, la estrategia de análisis, el desarrollo de modelos de ML y el acceso a los modelos deben diseñarse para manejar la mayor escala, el volumen de datos y los requisitos computacionales. Escalar la infraestructura, implementar marcos de cómputo distribuido y optimizar los pipelines para la automatización y eficiencia serían cruciales para satisfacer eficazmente las necesidades de la plantilla más numerosa. De manera más importante, quizás, una plantilla mucho más numerosa conllevaría la posibilidad de desarrollar modelos más precisos.

4 - Productivizar el modelo de *bank marketing*. Hay dos opciones para elegir: mediante el diseñador de Azure o bien de forma manual en su propia máquina (serializando los modelos/transformaciones con Pickle).

1. Si el objetivo del modelo es lograr predecir previo a que se realice la llamada qué personas tienen una alta probabilidad de contratar, ¿se podría decir si hay alguna variable que no sea posible incluir en el modelo en producción?, ¿Cuáles son?

La variable ‘duration’ está altamente relacionada con el target output (e.g. si duration=0, entonces y=no). La duración no se conoce antes de la llamada, y después se sabe la duración exacta. Como indican los autores del dataset, esta variable solo debería ser usada en benchmarking.
Se elimina también la variable ‘pdays’, puesto que contiene taje muy elevado de una sola categoría (‘-1’) y su poder decisorio es muy bajo, según el PCA realizado.

2. Probar varios modelos e ingeniería de características y dar con el modelo que mejor métrica presente.

Se adjunta un documento (ModeloML.pdf) detallando la elaboración del modelo y la ingeniería de características, así como el despliegue del endpoint.