MACHINE LEARNING ESCALABLE

PEC – CUESTIONARIO

MIGUEL PÉREZ

1. **¿Cómo deberían estar los datos sobre los que uso mi framework de ejecución paralela?**

Es necesario que donde se use el framework de ejecución paralela el dato esté distribuido o por lo menos particionado.

2. **¿Si tengo un proceso que se encarga recoger las señales de una serie de dispositivos de IOT, cual framework utilizarías?**

Se asume que las señales de los dispositivos de IOT son en tiempo real y que requerirán posiblemente de algún tipo de transformación, por lo que un buen framework en este escenario sería Apache Flink, siendo una herramienta que pone el foco en el procesamiento en tiempo real, proporcionando un alto rendimiento, tolerancia a fallos y la posibilidad de crear procesos end-to-end.

3. **¿Cuáles son los beneficios de usar los Spark ML Pipelines?**

Spark ML Pipelines es la herramienta de Spark MLib que permite diseñar flujos analíticos y ejecutarlos de manera end-to-end. De esta forma, se pueden especificar en un pipeline una secuencia de pasos, que se ejecutan en el orden en el que se añaden, y el conjunto de datos de origen se transforma a cada paso, de forma que la salida de un paso es la entrada del siguiente paso.

Esto permite diseñar flujos que se pueden guardar y cargar, y cuya entrada puede ser un set de datos en crudo al que se le puede aplicar todo tipo de transformaciones necesarias, tanto de limpieza como de creación de Features y salidas, para tener un Dataframe listo para un paso final que podría ser generar el modelo, o hacer una predicción, o también se podría incluir una evaluación. De esta forma, se abre una serie de posibilidades que permite generar flujos de manera sencilla y que es muy útil a la hora de programar en Spark.

4. **¿Si quisiera tener un entorno de python al 100% que framework de ejecución paralela podría usar?**

Un framework de ejecución paralela para tener un entorno 100% Python podría ser Dask, que nació en Python para Python.

5. **¿Me podrías decir y explicar algún otro framework de ejecución paralela, además de los que hayamos visto en clase?**

TensorFlow es una plataforma de código abierto de extremo a extremo para el aprendizaje automático. Cuenta con un ecosistema integral y flexible de herramientas, bibliotecas y recursos de la comunidad que les permite a los investigadores innovar con el aprendizaje automático y, a los desarrolladores, compilar e implementar con facilidad aplicaciones con tecnología de aprendizaje automático. Compila y entrena modelos de aprendizaje automático con facilidad mediante API intuitivas y de alto nivel como Keras, con ejecución inmediata, que permite una iteración de modelos inmediata y una depuración fácil, gracias a su uso de la computación en paralelo.

6. **¿Puedes explicar que tipo de servicio es Databricks (Iaas, PaaS, Saas ...etc.)?**

Databricks es del tipo de servicio Software as a Service (SaaS), que son aquellos que están diseñados para que puedas instanciar software de manera transparente a toda la plataforma que hay detrás e interactuar con él. Databricks está basado en Notebooks sobre AWS y lo que permite es el desarrollo de aplicaciones sobre Spark sin interactuar con lo que hay por debajo, lo que facilita la tarea del desarrollador.

7. **¿Me puedes dar algún ejemplo de DBaaS?**

DBaaS es un caso particular de SaaS, comentado en la pregunta anterior, para instanciar base de datos de forma fácil y rápida. Un ejemplo de ello es el servicio Azure SQL Database, que ofrece una base de datos totalmente gestionada con actualizaciones automatizadas, escalabilidad, protección inteligente contra amenazas y búsqueda potenciada por inteligencia artificial.

8. **¿Para un Data Science, que es más útil un IaaS o un PaaS?**

El IaaS está diseñado para poder solicitar servidores predefinidos sin necesidad de instalación de SO, ni configuración, ni pedir espacio en cabina, mientras que un PaaS está diseñado para poder solicitar plataformas de trabajo completas, es decir, además del servidor y SO, incluye una seria de productos/aplicaciones para poder trabajar, por lo que para un Data Scientist un PaaS es mucho más útil porque se ahorra la necesidad de instalar algunos productos, lo cuál puede llegar a ser muy tedioso.

9. ¿Ves algún riesgo en la cloud públicas?

Las cloud públicas presentan multitud de ventajas, pero también implican una serie de riesgos y limitaciones:

- Las máquinas virtuales están conviviendo en el mismo servidor con las de muchos otros clientes del proveedor, lo que puede llevar a que la actividad de un cliente pueda generar que el servidor fuera inhabilitado o causar algún otro problema.

- La gestión del sistema queda en manos del proveedor, lo que libera de carga al departamento de IT, pero es el proveedor quién toma las decisiones para mejorar la seguridad y calidad del servicio sin que el cliente tenga ningún poder de decisión.

- Posibles caídas de servidores.

- Posible dificultad para el cumplimiento de las normativas de seguridad de la información internas de la compañía.

- Mayor dificultad para realizar backups y restauración de datos.

- No hay control por parte del cliente sobre la seguridad.

- Posibles problemas regulatorios a la hora de alojar datos.

Muchos de estos problemas podrían evitarse mediante el uso de las cloud privadas, pero también hay que tener en cuenta que algunos de estos problemas vienen derivados de la falta de control de la empresa sobre los servicios, ya que es el proveedor quién lo gestiona, por lo que las cloud privadas dan mayor seguridad e independencia al cliente, pero también supone una mayor carga de trabajo para su departamento de IT, aunque, en mi opinión, es una mejor solución.

10. ¿Explícame en qué consiste el Data Version Control?

Un control de versiones es un sistema que registra los cambios realizados en un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante, y que permite la colaboración entre varios usuarios de un proyecto.

Por ejemplo, en desarrollo de aplicaciones o de proyectos de Machine Learning es ampliamente utilizado ya que permite la colaboración entre los distintos miembros de un equipo. La herramienta de control de versiones más común es Git. Un ejemplo de control de versiones aplicado a un proyecto de Machine Learning podría ser tener una rama *máster*, que pueda ser la de producción, una rama *desarrollo*, que pueda ser una

copia de la máster donde se realicen los test, y de dicha rama *desarrollo* cada desarrollador puede ir trabajando en diferentes ramas, desarrollando diversas partes del proyecto de forma que no se pisen el código, que cada uno pueda trabajar en su rama de forma local, y posteriormente su trabajo lo guarden en su rama del repositorio, de forma que el resto de desarrolladores tenga acceso a ello, y se lo pueda copiar a su rama si es necesario, y cuando esté listo, se puede copiar a la rama *desarrollo* para realizar los test pertinentes, y por último a la de producción.