Scalable Machine Learning on Big Data using Apache Spark

Week 1

Understanding how Apache Sparks works

1. Data storage solutions

Los conjuntos de datos pueden volverse bastante enormes y tener muchos esquemas diferentes. Porque para tener información relevante, es necesario almacenar datos durante bastante tiempo.

Opciones de almacenamiento:

- Datos SQL : tipo más común de sistemas de almacenamiento de datos. Compatibles con alta integridad y normalización de datos.
  Se puede acceder a los subconjuntos de datos a través de índices. Es un estándar abierto. El coste de almacenamiento es alto
  y además deben ser almacenamientos muy seguros.
- Datos NoSQL: no obligan a los datos a cumplir ningún esquema predefinido. Cada entrada puede tener su propio esquema.
  Son bases de datos lineales y escalables, si necesitan duplicar el almacenamiento duplican la cantidad de discos y listo.
  Pero esta flexibilidad tiene un precio.Por lo general, las bases de datos NoSQL no admiten la normalización o la integridad de los datos,
  por lo que los programas que interactúan con ellas tienen que cuidar esas propiedades si es necesario.
  Nunca alcanzaron el rendimiento de la consulta de la base de datos SQL.
- ObjectStorage: Sistemas de archivos remoto, escalables dinámicamente. Cantidad prácticamente ilimitada de capacidad de almacenamiento,
  y de alta disponibilidad integrada. Es linealmente escalable, por lo que básicamente solo almacena objetos de archivos sin necesidad de preocuparse la cantidad de datos que está almacenando, solo se le factura por cada gigabyte que consume cada mes.
- Datos IoT: proceden de una variedad de sensores y dispositivos y podrían implicar un amplio espectro de esquemas diferentes,
  que dependiendo del caso de uso, cambia rápidamente.

En caso de tener una cantidad baja de datos y un esquema muy estable se puede usar SQL.
A medida que aumenta la cantidad de datos, todo lo que tiene que hacer frente a esquemas que cambian continuamente, debe ir a NoSQL.
Finalmente, en cantidades muy altas de datos o esquemas muy inestables, incluso incluyendo audio, imágenes,
y datos de video, la elección principal es ObjectStorage.

En caso de que tenga que almacenar imágenes, audio y video, la opción principal es ObjectStorage, ya que todas las demás opciones implicarían una sobrecarga innecesaria.

2. Parallel data processing strategies of Apache Spark

Cuando hablamos de ApachesPark, hablamos de una máquina virtual Java. El motor de ejecución subyacente en ApachesPark está escrito en
un lenguaje de programación compatible con JVM.

Como regla general, una JVM por núcleo de CPU disponible.

Hay dos opciones para adjuntar a un clúster de ApachesPark:

- Opción 1: Usar un enfoque de almacenamiento fuera de nodo, donde el tercer sistema está conectado al clúster mediante una conexión de red rápida.
- Opción 2: Utilizar **switching fabric** que garantiza el máximo rendimiento de red entre sistemas de almacenamiento y nodos de trabajo. Principalmente
  usado cuando se requiere un alto ancho de banda.
- Otra opción es el enfoque JBOB (***Just a Bunch Of Discs***) es cuando se conectan directamente discos duros a los nodos de trabajo.

Consideremos un archivo, demasiado grande para caber en un solo disco. Lo que tenemos que hacer entonces es dividirlo en trozos de igual tamaño y
distribuirlos sobre los discos físicos.
Aquí es donde **HDFS** entra en juego. **HDFS** crea una vista virtual encima de esos fragmentos para que
pueda tratarse como un único archivo grande que abarca todo el clúster.
Una ventaja de la tecnología es la localidad de datos.Dado que HDFS es consciente de la ubicación de fragmentos individuales del archivo, los cálculos
se pueden realizar en paralelo utilizando CPU que residen en el mismo nodo de trabajo físico.

Resilient Distributed Dataset RDD Se trata de una colección o lista de datos inmutables distribuidos.
RDD se pueden escribir de manera que, por ejemplo, sólo contienen cadenas o valores dobles y
se crean a partir de datos existentes de diferentes fuentes como HDFS, ObjectStore, NoSQL o bases de datos SQL o simplemente archivos en un sistema de archivos local.

Una vez creado un RDD, reside distribuido en la memoria principal de los diferentes nodos de trabajo.
Cuando la memoria principal agregada de todos los nodos de trabajo no es suficiente, entonces los datos se dividen en disco.
Finalmente se puede decir que los RDD son "vagos". Eso significa que solo si los datos son realmente necesarios para
un cierto cálculo, se lee desde el sistema de almacenamiento subyacente, de lo contrario no.

Cuando se usa ApachesPark, programar es fácil, porque cuando se usa la API RDD simplemente no se pueden escribir programas no paralelos.
No importa cuántos datos proceses, el programa siempre permanece igual.
RDD proporciona la API central para lograr esto. Y no tiene que preocuparse por la distribución de datos y tareas.

ApachesPark se encargará de esto por ti.
Afortunadamente, usar ApachesPark no obliga a usar un lenguaje JVM como Scaler of Java.
También puede usar Python o R.