Comparación de H2O y MLib como herramientas para el aprendizaje automático en plataformas de Internet de las Cosas

H2O and MLib as tools for machine learning on Internet of Things platforms

Jorge Unger Rodríguez[1], Armando Jesús Plasencia Salgueiro[2]

Resumen

En este trabajo se compararon dos herramientas de aprendizaje automático, H2O y MLib, con el objetivo de seleccionar una de ellas para ser integradas en una plataforma de Internet de las Cosas. Para la comparación se tuvo en cuenta el tiempo de ejecución de las herramientas mientras ejecutaban el algoritmo K-means sobre un mismo conjunto de datos. En esta prueba H2O mostró los mejores resultados, y esto sumado a su fácil usabilidad y capacidad de integración con otras herramientas de aprendizaje automático lo hacen sobresalir sobre la gran variedad de opciones que existen para el aprendizaje automático.

Palabras clave: aprendizaje automático, Internet de las Cosas (IoT), H2O, Spark, MLib.

Abstract

In this work, two machine learning tools, H2O and MLib, were compared with the objective of selecting one of them to be integrated into an Internet of Things platform. For the comparison, the execution time of the tools was taken into account while executing the K-means algorithm on the same data set. In this test, H2O showed the best results, and this added to its easy usability and ability to integrate with other machine learning tools make it stand out on the wide variety of options that exist for machine learning.

***Keywords**: Machine learning, IoT, H2O, Spark, MLib*

[1] Universidad de la Isla de la Juventud "Jesús Montané Oropesa", Cuba, junger@uij.edu.cu
[2] Instituto de Cibernética, Matemática y Física (ICIMAF), Cuba, armando356@gmail.com

Introducción

Las plataformas de IoTgeneran una enorme cantidad de datos recogidos de diferentes tipos de dispositivos (sensores, teléfonos inteligentes) lo que implica la heterogeneidad de los datos. En este entorno de IoT además de la heterogeneidad se debe lidiar con fallos de conexión por lo cual los datos pueden estar incompletos o ilegibles. Otro desafío para el IoT es la necesidad de procesamiento en tiempo real; por lo que el análisis masivo de datos se hace necesario para lograr obtener conocimiento a partir de los datos generados por las distintas plataformas de IoT.

En este contexto del procesamiento de datos en IoT (Biswas, Dupont, & Pham, 2017) proponen el uso del aprendizaje automático, por su utilidad para descubrir conocimientos a partir de los datos generados. Los datos para el aprendizaje automático son generados a partir de una variedad de fuentes, tales como aplicaciones, sensores y dispositivos, estos son usados para el ajuste de los algoritmos, en la construcciónde modelos,para resolver un problema o generar conocimiento.(Gartner, 2017)

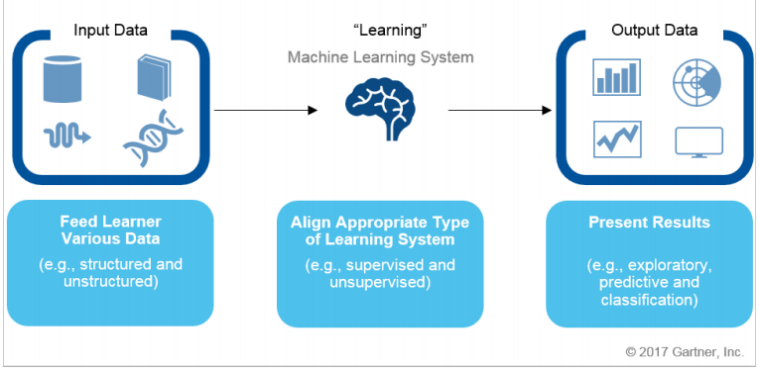

Figura1. Elementos Básicos del Aprendizaje Automático (Gartner, 2017)

El desafío de la comunidad de aprendizaje automático está en procesar de forma eficiente grandes volúmenes de datos (BigData). La proliferación del fenómeno BigData precisa a pensar no solo en las herramientas de procesamiento de datos sino también que nos obliga a tener en cuenta la implementación de los algoritmos de aprendizaje automático.(Landset, Khoshgoftaar, Richter, & Hasanin, 2015)

El objetivo de este trabajo es seleccionar una herramienta de aprendizaje automático a partir de una comparación entre ellas que permita distinguir cuál de estas es factible aplicar al desarrollo de una plataforma de Internet de las Cosas. La selección de este tipo de herramientas también ha sido tratada por otros autores como (Landset, Khoshgoftaar, Richter, & Hasanin, 2015), los mismos, comparan herramientas de aprendizaje automático con BigData en el ecosistema Hadoop; en el artículo se discuten las ventajas y desventajas de tres paradigmas de procesamiento a partir de la comparación de MapReduce, Spark, Flink, Storm y H2O. Otro trabajo relacionado es el de los autores (Gopalani & Arora, 2015) donde se compara el rendimiento de Apache Spark y MapReduce usando el algoritomo K-Means. Además de estos, otros autores como (Mahdavinejad, y otros, 2017) se centran en la presentación de una taxonomía de los algoritmos de aprendizaje automático que son adoptados por el IoT y en cómo aplicar estos algoritmos a los datos generados por el IoT.

Metodología

Para la selección del motor de procesamiento a emplear se tienen en cuenta: la latencia, el rendimiento, la tolerancia a fallos, la usabilidad y los recursos empleados. Como se observa en la Tabla 1 solo Spark y H2O sobresalen por cumplir con todos los requisitos, por lo cual se comparan las librerías de aprenidzaje automático MLlib de Apache Spark y H2O teniendo en cuenta el tiempo de ejecución del algoritmo K-Means.

Tabla 1 Motores de procesamiento de los datos para Hadoop(Landset, Khoshgoftaar, Richter, & Hasanin, 2015)

	Current stable release (as of June 1, 2015)	Execution model	Supported languages	Associated ML tools	In-memory processing	Low latency	Fault tolerance	Enterprise support
MapReduce	2.7.0	Batch	Java	Mahout	×	×	✓	×
Spark	1.3.1	Batch, streaming	Java, Python, R, Scala	MLlib, Mahout, H_2O	✓	✓	✓	✓
Flink	0.8.1	Batch, streaming	Java, Scala	Flink-ML, SAMOA	✓	✓	✓	×
Storm	0.9.4	Streaming	Any	SAMOA	✓	✓	✓	×
H_2O	3.0.0.12	Batch	Java, Python, R, Scala	H_2O, Mahout, MLlib	✓	✓	✓	✓

Destacar, que, de todos, H2O es el único que puede ser considerado un producto. Su característica más notable es que proporciona una interface gráfica (GUI) y numerosas herramientas para los algoritmos de aprendizaje profundo (deeplearning).(Landset, Khoshgoftaar, Richter, & Hasanin, 2015)

La investigación consiste en comparar el tiempo de ejecución de las librerías H2O y MLib corriendo el algoritmo K-Means sobre el mismo conjunto de datos.Para correr H2O se usó como lenguaje a R y para correr MLib se usó como lenguaje a Java.

Datos de prueba

Para realizar las pruebas se tiene el set de datos "Wine". Estos datos son resultados del análisis químico de distintos vinos cultivados en la misma región de Italia paro derivados de tres diferentes cultivos. Existen trece atributos y tres clases, de la clase 1 hay 59 instancias, de la clase 2 hay 71 instancias y de la clase 3 hay 48 instancias.

Fuente de los datos: Forina, M. et al, PARVUS - AnExtendiblePackagefor DataExploration, Classification and Correlation. Institute of PharmaceuticalandFoodAnalysis and Technologies, ViaBrigata Salerno, 16147 Genoa, Italy.

Hardware para la prueba

Para realizar la prueba se tiene una laptop con procesador AMD A6-3420M APU con Radeom HD GraphicsQuadCore 1.50 GHz, memoria RAM de 4,00 GB y sistema operativo Windows 7 UltimateService Pack 1 de 64 bits.

Herramientas para la prueba

Tabla 2 Versiones de software empleado

	Versión
H2O	3.16
Apache Spark	2.2.1

MLib	2.11
Java	1.8
R	3.4.3

Resultados y discusión

En el entorno de investigación o de producción, la selección de la librería de aprendizaje automático o de un algoritmo está en dependencia de varios factores, mayormente depende de las necesidades de un determinado grupo o proyecto. Para evaluar las herramientas de aprendizaje automático se tuvieron en cuenta las siguientes consideraciones:

Velocidad: Lo que más afecta a la velocidad es sobre cuál plataforma de procesamiento está corriendo la librería o el algoritmo. (Landset, Khoshgoftaar, Richter, & Hasanin, 2015)

Usabilidad: Es tratada en términos de dificultad de instalación, lenguajes de programación disponible, disponibilidad de una interfaz de usuario, cantidad de documentación disponible.(Landset, Khoshgoftaar, Richter, & Hasanin, 2015)

En laFigura 2 y Figura 3 se muestra el fragmento de código ejecutado en R y en Java para H2O y MLib respectivamente. En ambos casos solo se evalúa el tiempo que demora cada librería en crear el modelo.

```
old <- Sys.time() # get start time

wine.km = h2o.kmeans(wine.hex, k = 3)
print(wine.km)

new <- Sys.time() - old # calculate difference
print(new) # print in nice format
```

Figura 2 Fragmento de Código en R

```
long startTime = System.currentTimeMillis();
KMeansModel clusters = KMeans.train(parsedData.rdd(), numClusters, numIterations);
long endTime = System.currentTimeMillis();
long TotalTime = endTime - startTime;
System.out.println("Tiempo de Ejecucion "+TotalTime);
```

Figura 3 Fragmento de Código en Java

Durante las pruebas (ver Tabla 3) el desempeño de H2O superó a MLib. El tiempo de ejecución promedio de H2O fue de 0.347 segundos mientras que el tiempo de ejecución promedio de MLib fue de 3.531 segundos. No obstante, se debe tener en cuenta que MLib está corriendo sobre Java y esto puede afectar su rendimiento por lo que el resultado de esta prueba no es absoluto.

Tabla 3. Comparación del tiempo de ejecución del algoritmo K-Means

	Tiempo de ejecución#1	Tiempo de ejecución#2	Tiempo de ejecución#3	Tiempo de ejecución#4	Tiempo de ejecución#5
H2O	0.365 s.	0.342 s.	0.377 s.	0.337 s.	0.314 s.
MLib	4.083 s.	3.743 s.	3.143 s.	2.672 s.	4.016 s.

Conclusiones

En el desarrollo de una plataforma de Internet de las Cosas se debe tener en cuenta la masividad y heterogeneidad de los datos con los que esta debe lidiar. Extraer conocimiento de forma eficiente a partir de estos datos requerirá de una arquitectura compleja que combine herramientas y técnicas para la recopilación, el almacenamiento, el procesamiento y el análisis. En este trabajo H2O muestra mejor desempeño que MLib en cuanto a usabilidad y velocidad; no obstante, lo sobresaliente de H2O es su capacidad de integrarse tanto con MLib como con otras librerías tales como Mahout de aprendizaje automático y es esto último lo que hace a H2O el mejor candidato para el aprendizaje automático en una plataforma de Internet de las Cosas.

Bibliografía

Biswas, A. R., Dupont, C., & Pham, C. (2017). IoT, Cloud and BigData Integration for IoT Analytics. En J. Soldatos, *Building Blocks for IoT Analytics* (pág. 294). River Publishers.

Gartner. (2017). *Preparing and Architecting for Machine.*

Gopalani, S., & Arora, R. (2015). Comparing Apache Spark and Map Reduce with Performance Analysis using K-Means. *International Journal of Computer Applications*, 1-4.

Landset, S., Khoshgoftaar, T., Richter, A., & Hasanin, T. (2015). A survey of open source tools for machine learning with big data in the Hadoop ecosystem. *Journal of Big Data*, 1-36.

Mahdavinejad, M. S., Rezvan, M., Barekatain, M., Adibi, P., Barnaghi, P., & Sheth, A. (2017). Machine Learning for Internet of Things Data Analysis: A Survey. *Digital Communications and Networks*, 1-56.