CSE 598 – Data Intensive Systems for Machine Learning

Assignment – 8 Report

Submitted by Uma Mageswari Rajendiran (1217196084)

I. Hardware Configuration:

Processor	Intel Core i9
Base Clock Frequency	2.3 GHz
Number of Cores	8
Memory	32 GB 2667 MHz DDR4
Graphics	AMD Radeon Pro 5500M 8 GB Intel UHD Graphics 630 1536 MB

II. Software Configuration:

Operating System	macOS Big Sur 11.2.3
Java	15.0.2
Scala	2.11.12
Spark-core	2.4.5
Spark-mllib	2.4.5
Breeze	0.11.2
Sbt	1.5.0

III. Performance Comparison:

The following tables shows the performance comparison of Handcoded Scala, Spark and Breeze for 100, 500 and 1000 iterations respectively,

A. 100 Iterations:

	Handcoded Scala	**Spark**	**Breeze**
Iterations	100	100	100
Avg Time Taken	163.708 s	0.86 s	0.89 s

B. 500 Iterations:

	Handcoded Scala	Spark	Breeze
Iterations	500	500	500
Avg Time Taken	930.582 s	4.363 s	4.575 s

C. 1000 Iterations:

	Handcoded Scala	Spark	Breeze
Iterations	1000	1000	1000
Avg Time Taken	2085.248 s	8.734 s	9.290 s

IV. Observation:

- Hand-coded Scala was the slowest for 100, 500 and 1000 iterations.
- Spark was the fastest for 100, 500 and 1000 iterations.
- Both Spark mllib's and Breeze's DenseMatrix provided much better results for 100, 500 and 1000 iterations in comparison to the hand-coded Scala.
- The difference in average time taken between Spark and Breeze for 100, 500 and 1000 iterations are within the margin of error.
- The performance difference between using Spark and Breeze is negligible.

V. Lessons Learned:

- It is expensive to implement numerical operations on large size dense vectors.
- We can experience huge performance gain with the help of libraries like Breeze and Spark, as their implementations for linear algebra operations are highly optimized.
- I learned and gained some hands-on experience with Scala and the sbt build tool.
- I tried compiling the code with sbt to create a fat jar as spark-submit accepts only fat jars.
- I realized that creating thin jars on sbt was easy. However, creating fat jars using sbt was hard as I had to figure out merge strategies for conflicting packages during assembly and there were lots of versioning dependencies.
- For building packages, I found Maven and Gradle to be much easier to work with.