**BIG DATA COMPUTING 2021/22 - PROJECT 3**

**JAVA VERSION**

Run your algorithm on the cluster on CloudVeneto using the following datasets: **HIGGS10M7D.txt** (about 10M points in 7 dimensions),and **artificial9000.txt** (9200 points in 2 dimensions).The datasets are in the **directory /data/BDC2122** of the HDFS. You must fill the two tables below, one for each dataset, where the headers of the rows indicate the values to report, and the headers of the columns indicate the configurations of parameters to be used.

The first table collects results aimed at assessing the **scalability** of the algorithm.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **HIGGS10M7D.txt** | **2 executors**  **k=10, z=150, L=2** | **4 executors**  **k=10, z=150, L=4** | **8 executors**  **k=10, z=150, L=8** | **16 executors**  **k=10, z=150, L=16** |
| **Time to read input from file (in ms)** | 46888 | 30730 | 19594 | 12166 |
| **Time of ROUND 1 (in ms)** | 40574 | 21189 | 12085 | 5949 |
| **Time of ROUND 2 (in ms)** | 59 | 129 | 247 | 658 |
| **Time to compute objective function (in ms)** | 20749 | 9518 | 5582 | 4441 |
| **Value of objective function** | 11.48421725 | 10.49952379 | 9.286101436 | 8.607740371 |

The second table collects results aimed at comparing the **accuracy** attained by the algorithm against the one attained by the sequential algorithm from Homework 2 on the entire dataset.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Artificial9000.txt** | **2 executors**  **k=9, z=200, L=2** | **4 executors**  **k=9, z=200, L=4** | **8 executors**  **k=9, z=200, L=8** | **16 executors**  **k=9, z=200, L=16** | **Sequential algorithm from Homework 2 with k=9 and z=200** |
| **Value of objective function** | 12.71808778865754 | 12.620865421990683 | 12.269031950402614 | 11.799656139057612 | 11.576939707884812 |

Provide below a brief comment to justify the scalability and accuracy observed (your answer should be of at most 6 lines, font 12 points):

The data from the MapReduce implementation shows that using more executors results in better efficiency. All operations that make use of partitioning: file read, round 1 and objective function, show consistently better running times when increasing the executors and partitions, while the sequential computation in round 2 grows but remains low thanks to the small size of the coreset. Accuracy tends to increase with the partitions, given a larger coreset, but it is still a moderately worse approximation compared to the sequential algorithm run on the entire dataset.