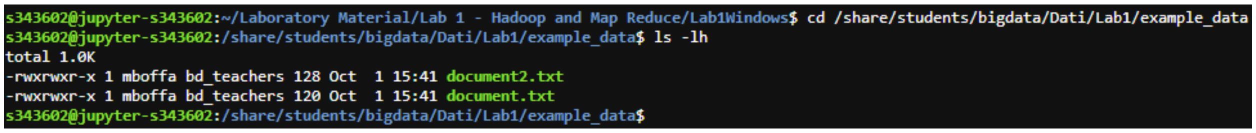
Lab 1 - Hadoop and Map Reduce

3. Explore the BigData@Polito

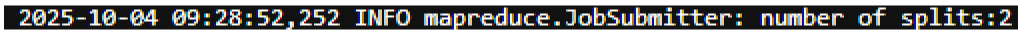
What's the content of the `/share/students/bigdata/Dati/Lab1/example_data` folder? What is the size of each line?



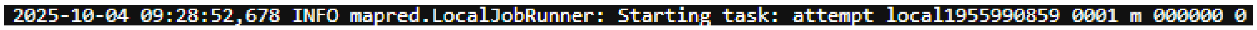
```
s343602@jupyter-s343602:~/Laboratory Material/Lab 1 - Hadoop and Map Reduce/Lab1Windows$ cd /share/students/bigdata/Dati/Lab1/example_data
s343602@jupyter-s343602:/share/students/bigdata/Dati/Lab1/example_data$ ls -lh
total 1.0K
-rwxrwxr-x 1 mboffa bd_teachers 128 Oct  1 15:41 document2.txt
-rwxrwxr-x 1 mboffa bd_teachers 120 Oct  1 15:41 document.txt
s343602@jupyter-s343602:/share/students/bigdata/Dati/Lab1/example_data$
```

5. Submit a job

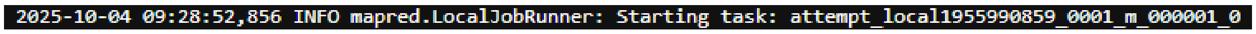
Check the number of mappers (i.e., the number of instances of the mapper class)



```
2025-10-04 09:28:52,252 INFO mapreduce.JobSubmitter: number of splits:2
```

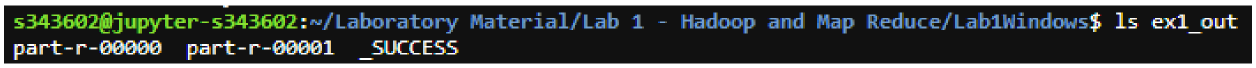


```
2025-10-04 09:28:52,678 INFO mapred.LocalJobRunner: Starting task: attempt_local1955990859_0001_m_000000_0
```



```
2025-10-04 09:28:52,856 INFO mapred.LocalJobRunner: Starting task: attempt_local1955990859_0001_m_000001_0
```

What is inside `ex1_out`? How many files do you see? Why do you think it is the case?



```
s343602@jupyter-s343602:~/Laboratory Material/Lab 1 - Hadoop and Map Reduce/Lab1Windows$ ls ex1_out
part-r-00000  part-r-00001  _SUCCESS
```

There are 3 files, because 2 reducers were used. Hadoop launched `r_000000` and `r_000001`, so it wrote two output files: `part-r-00000` and `part-r-00001`. The third file `_SUCESS` is an empty marker created by the file `FileOutputCommitter`, which states that the task ended without errors.

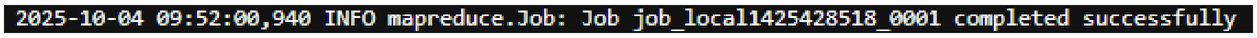
Try to re-run the same job. Does it succeed this time? If not, what is the problem?

No, because the output folder already exists. Hadoop does not overwrite the output and throws `FileAlreadyExistsException`.

Remove `ex1_out` (`rm -r ./ex1_out`) and re-run now. Does it work?

Yes.

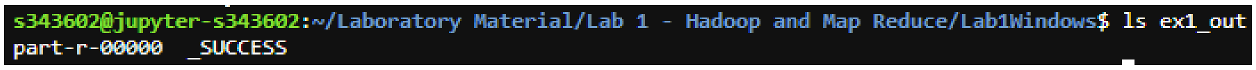
There are some evidences in logs:



```
2025-10-04 09:52:00,940 INFO mapreduce.Job: Job job_local1425428518_0001 completed successfully
```

- Two reduce tasks executed and committed (`r_000000`, `r_000001`).
- Output wrote in `ex1_out` with `Bytes Written=65` and `75`. Total groups reduced: `7 + 8 = 15`.

Run again the application and change the number of reducers (first parameter of the application, set it to 1). Analyze the content of the output folder. How many files do you see now?



```
s343602@jupyter-s343602:~/Laboratory Material/Lab 1 - Hadoop and Map Reduce/Lab1Windows$ ls ex1_out
part-r-00000   _SUCCESS
```

6. Test on a bigger file

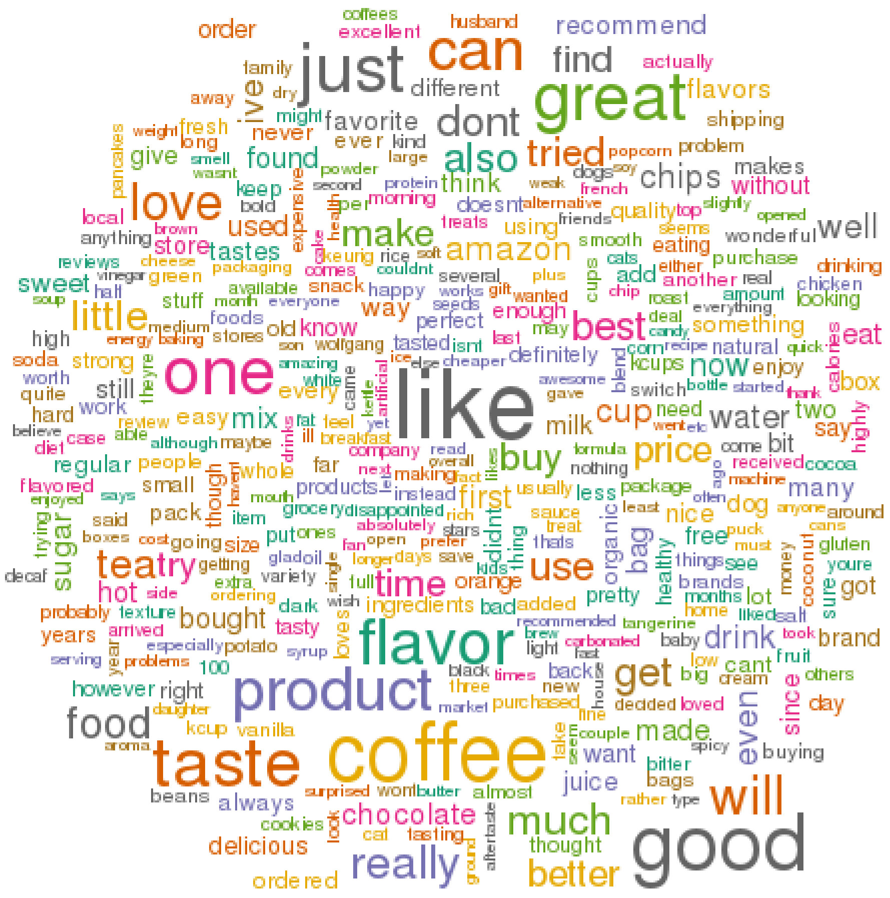
Analyze the results

Can you understand any interesting facts from your results?

- Job scale: 1 file, 8 split of ~32 MiB each → 8 maps and 2 reduces. Execution LocalJobRunner (prefix `job_local`).

- Volume: ~244 MB read, 46 013 295 token produced by mappers, 286 173 unique keys (vocab size).
  Average ≈ 160.8 occurrences for word (46 013 295 / 286 173).
- Heavy shuffle: ~460.6 MiB transferred to reducers (`Reduce shuffle bytes=514,424,937`).
  Spilled Records=136 260 955 → more sort/merge steps.
- Moderate skew on reducers:
  R1 input 26.83 M records, R2 19.19 M (≈58%/42%), but similar output groups (142 984 vs 143 189).
- Output: ~3.35 MB total written, two files `part-r-0000{0,1}` + `_SUCCESS`.
- No combiner: `Combine input/output records=0` → missed chance of reducing shuffle.

The following figure was done on a small sample of your data (10000 reviews).



Is it consistent with what you found on the complete dataset?

Yes, the figure is coherent with the complete dataset, because the job processed 568 454 total reviews. There are 286 173 distinct words. About 46 M token were emitted. About 81 token for every review. These numbers are compatible with long reviews of “Amazon Fine Foods”.

In the figure, there are some expected terms: “coffee”, “tea”, “chocolate”, “chips”, and generic adjectives/opinions: “good”, “great”, “taste”, “like”, “love”, “product”, “buy”. These are top-token of the corpus.

Both reducers wrote 2 files. The final reduction to ~286 k rows suggest a classical word-count. No particular anomaly.

Do you think a small sample is enough to represent the whole?

Yes, but only for frequent patterns. Not for the part of the distribution of words with many rare terms.

7. Bonus track

Modify your count `.java` program to count 2-grams frequencies. Consider each line as a separate document, as in the Amazon reviews file (so, do not count as contiguous words on separate lines).

How long would you expect it to run, compared to the simple word count?

~+20% compared to the running time of simple word count.

- Simple word count: 2:53
- 2-grams: 3:28