ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Scuola di Ingegneria e Architettura
Laurea Magistrale in Ingegneria e Scienze Informatiche

USED CARS ANALYSIS

Elaborato in
BIG DATA

Presentata da
GABRIELE GUERRINI

Anno Accademico 2019 – 2020

Table of contents

Chapter 1

Dataset and query

1.1 Dataset

The dataset can be downloaded at:
https://www.kaggle.com/austinreese/craigslist-carstrucks-data.

1.2 Query

The query to be executed is the following one:

"For each region, it must be found the OPI of the most widespread brand in such region, considering cars that use gas fuel only.

OPI is an acronym for Odometer-Price Index and represents the average ratio odometer/price. It is calculated upon all cars of a given brand in the country, regardless of other car features (fuel type, number of cylinders...)."

Chapter 2

MapReduce

2.1 Workflow

Figure 2.1 shows the adopted workflow[1].

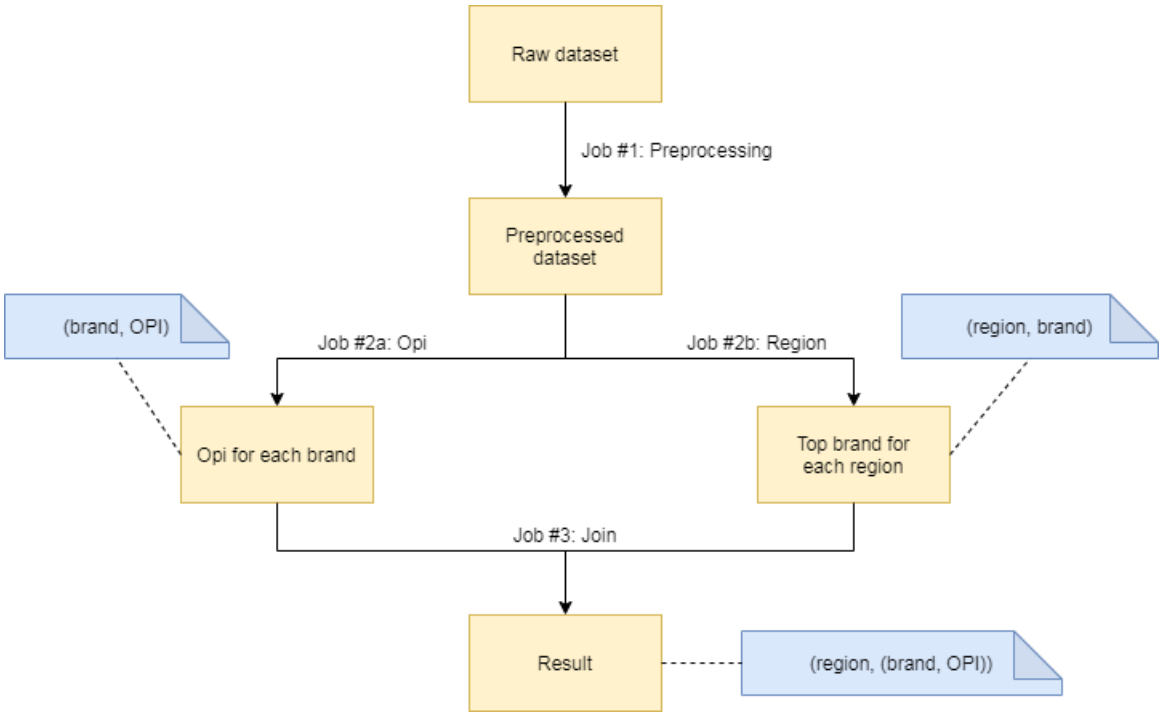

Figure 2.1: Adopted workflow for MR jobs.

[1]It is not the optimal one since the same query could be executed in less than four job. The idea is to structure the workflow this way so that few different MR algorithms can be applied (e.g. filtering, projection, summarization, join).

2.2 Jobs

2.2.1 Job 1: Preprocessing

The job executes all preprocessing operation needed to correctly use the dataset in next jobs. In particular, it fulfills the following goals:

- **Cut out header**: the raw dataset is a csv file. The header must be eliminated.

- **Drop useless columns**.

- **Drop incomplete records**: record that have missing values on mandatory fields are simply dropped.

  Job execution and interfaces are described in figure 2.2.

  Raw records are read and parsed into "Car", i.e. custom "Writable" objects, during map stage. Each car stores data about:

  - Region
  - Price
  - Brand
  - Fuel
  - Odometer

  The map output is a pair where the key is the default key used by Hadoop when reading text files and the value is the car itself.

  The reduce stage just replaces the default key with a "NullableWritable" so that the whole job output is a set of records yet.

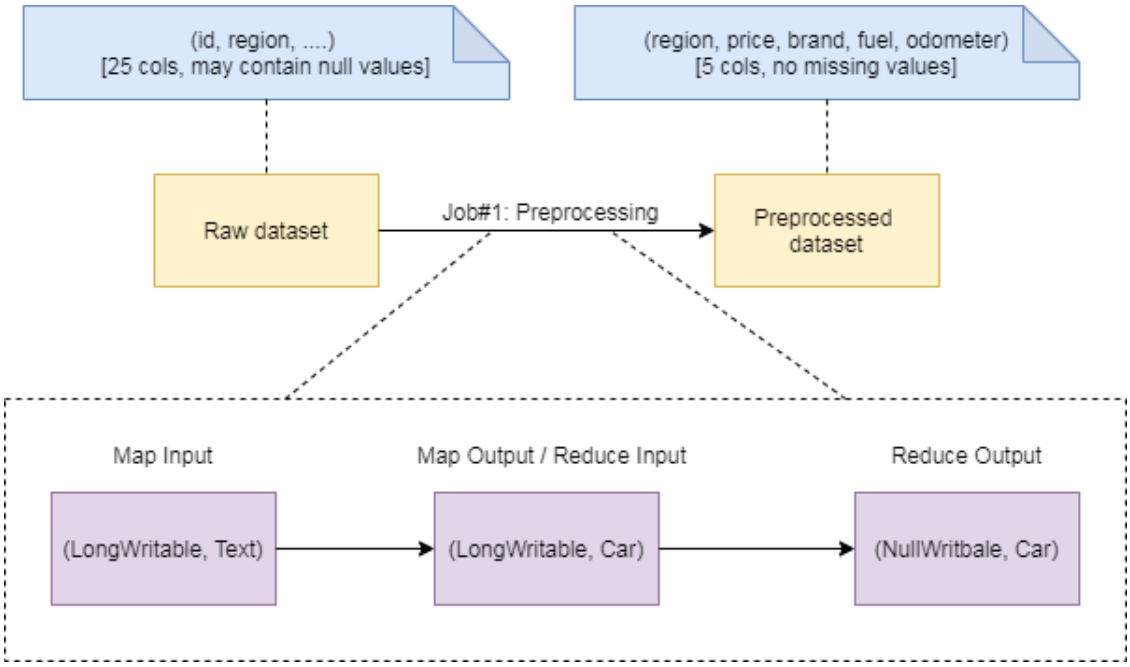

Figure 2.2: Description of "job 1: preprocessing" workflow and interfaces.

2.2.2 Job 2a: Opi

The job executes the calculation of the OPI for each existing brand.

Job execution and interfaces are described in figure 2.3.

The map stage input value is a car encoded as text. The key is the Hadoop default one for text files and its irrelevant. The mapper calculates the OPI for such car using price and odometer fields, and it gives as output a pair consisting of (brand, OPI).

The reduce stage just calculates the average value using single OPI values and reports it as output.

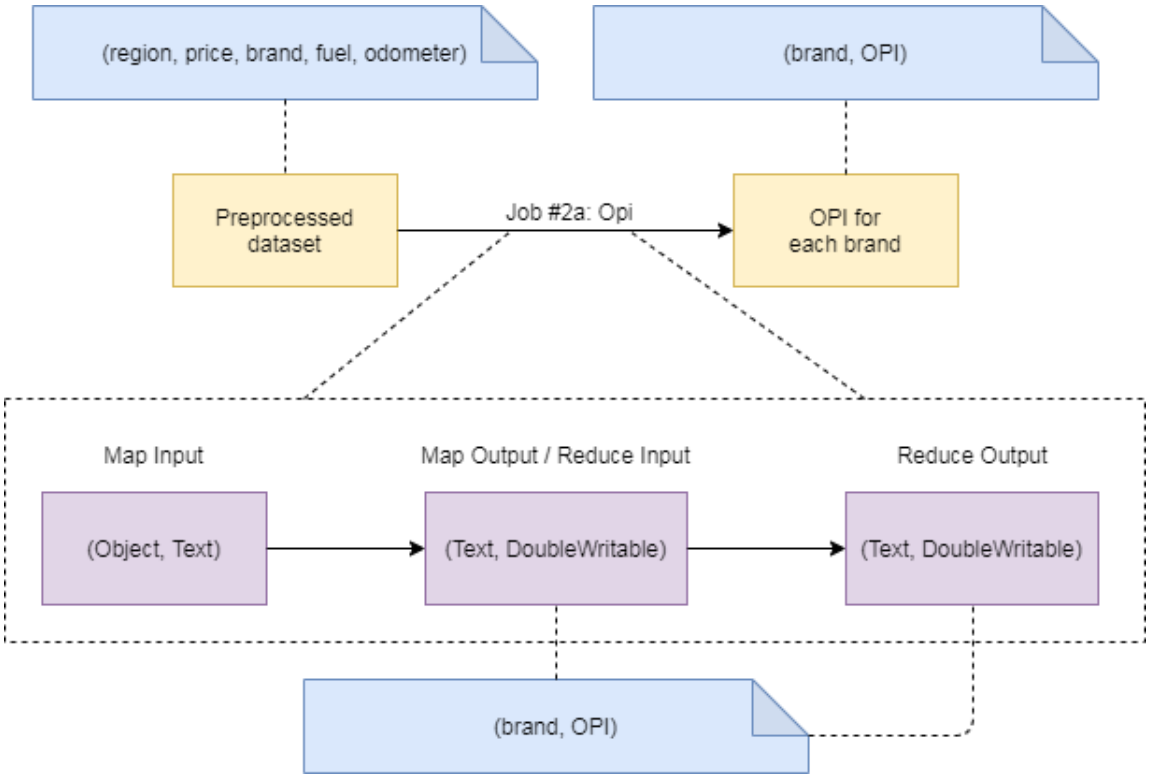

Figure 2.3: Description of "job 2a: opi" workflow and interfaces.

2.2.3 Job 2b: Region

The job goal is to find the most widespread brand in each region, namely the brand that has the most number of car on sale in a given region.

Job execution and interfaces are described in figure 2.4.

The map stage input value is a car encoded as text. The key is the Hadoop default one for text files and its irrelevant. If the fuel type is correct, the mapper generates as output a pair (region, brand) so that reduce stage can operate per region basis.

The reducer, given a region, firstly calculates the cardinality for each brand by collecting the input pairs. Then, it calculates the maximum and gives it as output in the form of (region, brand).

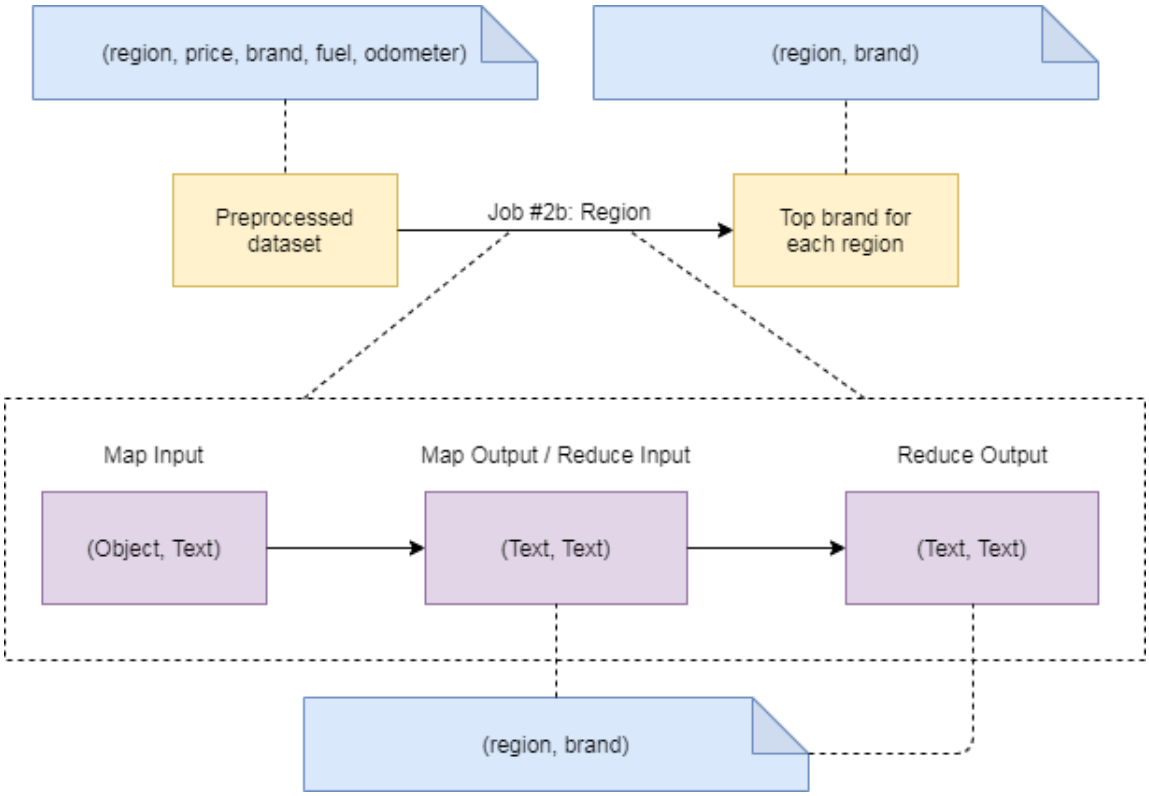

Figure 2.4: Description of "job 2b: region" workflow and interfaces.

2.2.4 Job 3: Join

Partial results of jobs 2a and 2b are combined into the final one.

Job execution and interfaces are described in figure 2.5.

The map stage takes as input pairs structured as (Text, Text). The source of a given pair can be from job 2a output or from job 2b output. By examinating it, it is discovered the source and the pair is marked with a flag. Since the join key will be the brand, the output of map stage is a pair (brand, (flag, region)), where flag and region are encapsulated into a "JoinPair" (a custom "Writable").

The reducer executes the standard join algorithm on input items and creates output pairs in the form (region, (brand, OPI)).

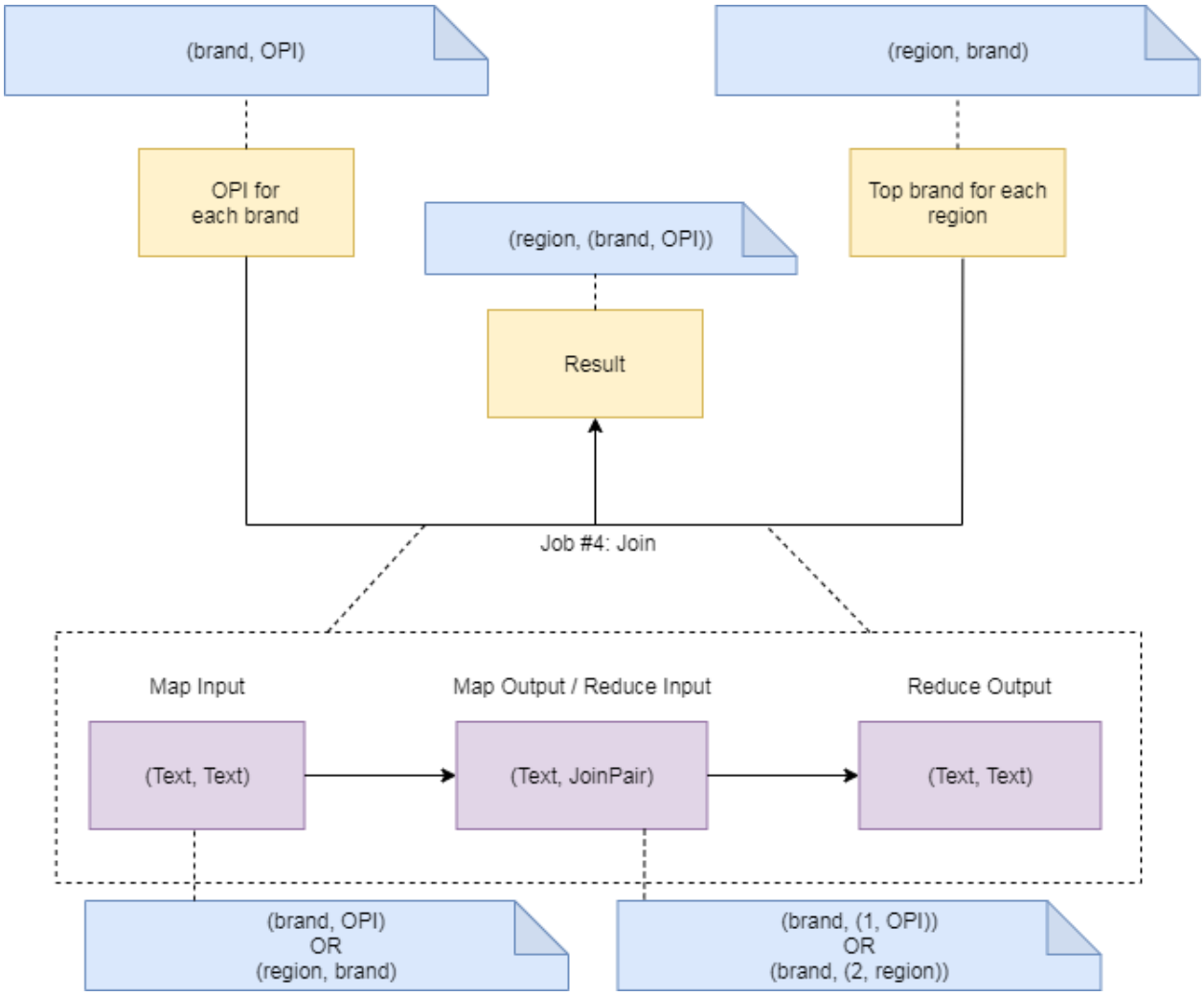

Figure 2.5: Description of "job 3: join" workflow and interfaces.

2.3 Performance evaluation

Performance are evaluated by varying the number of reducers and by enabling the use of combiners.

Each casuistry result is the average value upon 3 runs.

		Reducers				
		1	2	5	10	100
Combiner	No	223	162	168	216	1400
	Yes					

Table 2.1: Performance results expressed as elapsed time in seconds.

Chapter 3

Spark

3.1 Workflow

3.2 Performance evaluation

Chapter 4

Conclusions