Università degli Studi di Milano

Data Science and Economics

Algorithms for Massive Datasets
Market-Basket analysis

Luca Perego
Registration numbers: 26590A

ACADEMIC YEAR 2023-2024

Contents

Luca Perego

September 2024

1 Introduction

The *market-basket* model describes many-to-many relationships between two objects. These objects are *items* and *baskets*. A *basket* can be interpreted as an observation.
The general assumption is that the number of baskets is very high, to the point that the file containing the baskets cannot be loaded in memory. The items are the elements within each basket, and their overall number is assumed to be lower than the absolute number of baskets. An intuitive example is provided by the retail sector, where the *items* are the elements sold by a retailer, while the *baskets* are the transactions carried out by customers.

2 Dataset, Preprocessing and EDA

2.1 Dataset information

The dataset employed is the "LinkedIn Jobs and Skills 2024" obtained from Kaggle.
The data is downloaded from the Kaggle API directly through GoogleColab.
Although the dataset is comprised of three csv files, the analysis revolves just around `job_skills.csv`.
By running the code, the unnecessary files are automatically removed.

`job_skills.csv` is organized in two columns, one for the *job link* and one for the *job skills*. The column of interest is the latter, which is composed of many strings, each representing a skill associated with the job posting.
The overall dataset dimensions are: (1296381, 2)

2.2 Preprocessing

Since the analysis focuses on the *job_skills* column, the *job_link* column is dropped.
The NAs are dropped, although there were only 2007 of them. Then, the data is associated to a Resilient Distributed Dataset (RDD) through `PySpark`. Thus, the data is divided in 6 partitions.
To implement the market-basket analysis the data must be in sets containing the items, possibly the items should be represented by integers. To achieve this, the following is done:

1. We start from a Spark Dataset called `df_rdd`
2. The RDD object is mapped to a pipelined RDD through
   `skills = df_rdd.map(lambda x:  x['job_skills'])`

3. The baskets are created by first transforming all skills in lowercase and then splitting each item when a comma is found:
   ```
   skills = skills.map(lambda word:  word.lower())
   skills = skills.map(lambda line:  line.split(', '))
   ```
4. The list of all unique skills is obtained
5. An hash table is created, linking each skill with an integer
6. Baskets are created by translating each skill with the corresponding integer.

For example, the following basket:

```
['building custodial services',
'cleaning',
'janitorial services',
'materials handling',
'housekeeping',
'sanitation',
'waste management',
'floor maintenance',
'equipment maintenance',
'safety protocols',
'communication skills',
'attention to detail',
'physical strength',
'experience in housekeeping']
```

becomes:

```
{0,
1,
2,
462123,
462124,
923331,
1385287,
1385288,
1847540,
1847541,
1847542,
1847543,
1847544,
2309149}
```

Note that at the end of the process each basket is represented by a *set*.

2.3 Exploratory Data Analysis

The data is briefly analysed and the following facts are discovered:

- There are 1287105 unique values of job skills
- There are 2007 NAs in job skills
- The minimum number of skills required for a job posting is 1
- The maximum number of skills required for a job posting is 463
- The distribution of the number of skills required for a job posting is the following:

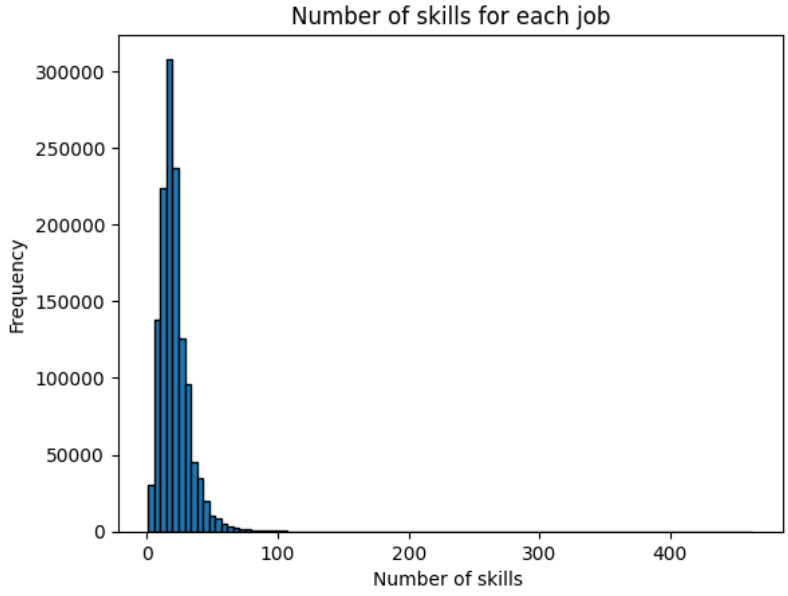

Figure 1: The overwhelming majority of job postings has between 1 and 100 required skills, with a peak around 30 skills. The plot highlights that there are outliers, for instance the maximum value that appears to be 463.

3 A priori algorithm implementation

The A priori algorithm is an efficient way to find the frequent itemsets in market-basket analysis. It is considered a baseline, on top of which more complex algorithms are developed.

Before starting with the algorithm, a "frequency threshold" must be defined. An itemset will be considered "frequent" if its number of occurrences is above such threshold. In this project an itemset is considered frequent if it appears in 2% of all baskets. Therefore, an itemset to be frequent

must appear in **at least 25887 job postings**.

The algorithm works in the following way:

1. All single items are counted, the ones appearing more than the threshold value are considered "frequent singletons". More specifically:
   - The baskets are flattened, creating for item a tuple of the type `(skill_integer, 1)`.
   - All the newly formed tuples are aggregated with a `.reduceByKey` operation, through which all the 1s (the second element of the tuple) are counted.
   - The skills with a count above the threshold are filtered.
2. All the pairs of frequent singletons are computed through the `combinations` function from the `itertools` package. These are the "candidate frequent pairs".
3. The occurrence of all the candidate frequent pairs is computed. The pairs whose count is above the threshold are the frequent pairs.
4. The algorithm increases the size of the itemset of interest until there are no more frequent itemsets.

This specific implementation of the A-Priori algorithm employes the complete dataset. For each pass (e.g. stage during which the algorithm iterates over the set of candidates itemsets) the number of frequent itemsets is provided and the top 3 most frequent itemsets are printed.

4 Results and discussion

4.1 Results

The following results are obtained:

- FIRST STAGE
  - 80 frequent itemsets of size 1 are found
  - The three most frequent singletons and their respective counts are:
    - `communication : 365844`
    - `customer service : 276404`
    - `teamwork : 225953`
- SECOND STAGE
  - 67 frequent itemsets of size 2 are found
  - The three most frequent pairs and their respective counts are:
    - `('teamwork', 'communication') : 139150`
    - `('customer service', 'communication') : 138992`
    - `('leadership', 'communication') : 117141`

- THIRD STAGE
  - 25 frequent itemsets of size 3 are found
  - The three most frequent triples and their respective counts are:
    * `('teamwork', 'customer service', 'communication') : 63922`
    * `('teamwork', 'problemsolving', 'communication') : 51228`
    * `('teamwork', 'leadership', 'communication') : 50878`
- FOURTH STAGE
  - No frequent itemsets of size 4 are found.

4.2 Discussion

With a frequency threshold of 2% we find the following frequent itemset:

- 80 frequent singletons
- 67 frequent pairs
- 25 frequent triples

The proposed solution should scale easily with data size. Thanks to the use of Spark's RDD and pipelines, the whole dataset could be employed in the analysis. Overall, it takes around 20 minutes to run the whole program (dependencies' installation included), and around 13 minutes for the A Priori algorithm to examine all itemsets. However, different solutions for market-basket could be investigated (such as the PCY algorithm). Additionally, a solution to drastically reduce computation time is sampling the data.

5 Declaration

I/We declare that this material, which I/We now submit for assessment, is entirely my/our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my/our work, and including any code produced using generative AI systems. I/We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me/us or any other person for assessment on this or any other course of study.