Algorithms for massive datasets

Market Basket Analysis

Riccardo Tenuta 26940A

A.Y. 2023/2024

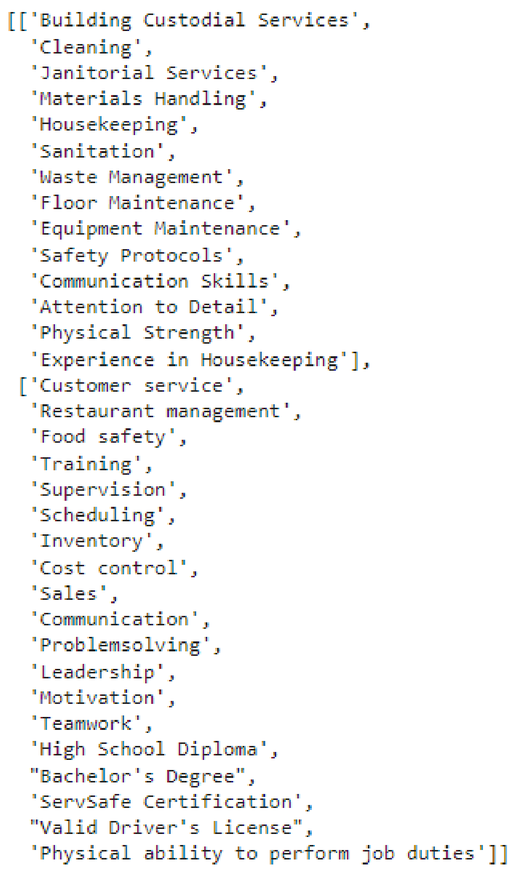
Introduction

The goal of the project is to implement the market basket analysis to a dataset containing information about the job offer market. The following document will describes all the steps and the result of the analysis starting from the data description and preparation to the computation of the SON (Savasere, Omiecinski, and Navathe) algorithm and of the association rules between the different frequent itemsets. The market basket analysis has the goal to find frequent itemsets between the baskets where every basket represents in this case a set of words/skills which are required for a specific job offer. This kind of analysis, including also the retrieve of the association rules, is particularly useful to find relations between the skillset in the job market.

1 Data preparation and pre-processing

The dataset *job_skills.csv* has been imported in the notebook but only the column *job_skills* column has been considered for the analysis. For the market basket analysis we defined as baskets the group of skills assigned to each job offer and an item as a unique value within a basket. After importing the dataset the first step was to remove any null value from the dataset. The second step has been the split of each basket (still represented as a *string*) in a *list* of separated values/skills that will be used to iterate over during the computation of the algorithm. The computation is executed using Apache Spark, a framework for distributed and parallel compunting specially used in analytics and with the execution of a CPU-intensive algorithm such as in our case with the SON algorithm. Because of that all the imported baskets are stored in a RDD (Resilient Distributed Dataset) created with the **pyspark** python library.
The 2% of the dataset has been sampled to compute the algorithm in a reasoned time, the same percentage has been set for the support threshold over the whole dataset. Based on the Apache Spark Organization the optimal number of partitions for each CPU core is between 2 and 4, so since the data are not stored on a DFS (Distributed File System) all the computation is executed on a single node splitting the latter on different processes. The ultimate number of partition is therefore 4 multiplied by the number of CPU cores available on the machine. Since the support threshold is

2% of the whole dataset for each partition/chunk of the dataset the support threshold will be 85 (the overall support threshold divided for the number of partitions).



```
[['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'],
 ['Customer service',
  'Restaurant management',
  'Food safety',
  'Training',
  'Supervision',
  'Scheduling',
  'Inventory',
  'Cost control',
  'Sales',
  'Communication',
  'Problemsolving',
  'Leadership',
  'Motivation',
  'Teamwork',
  'High School Diploma',
  "Bachelor's Degree",
  'ServSafe Certification',
  "Valid Driver's License",
  'Physical ability to perform job duties']]
```

Figure 1: First two baskets after the data preparation

2 The SON algorithm

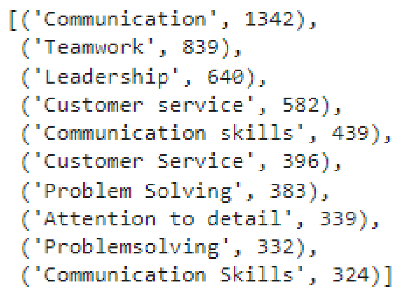
To perform the computation of the frequent itemsets it has been selected the SON algorithm which idea is to parallelize the computation over a set of chunks and to avoid both false negatives and false positives. After the chunking the simple, randomized algorithm is executed on each chunk/partition to extract the candidate itemsets. Eventually all the candidate itemsets for each partition were grouped and split again between the partitions in order to calculate the final and real support of each itemsets. This last pass is fundamental when at the end of this analysis the association rules have been extracted. During the execution multiple-size itemsets have been considered iteratively and incrementally and so to form sets with 1, 2, 3 and so on number of unique elements until the algorithm could not found anymore frequent itemsets in any available partition. Initially the first two passes have been explore in depth in order to better describe the functioning of the SON algorithm.

3 Scalability of the solution

Considering the potential of the SON algorithm in terms of parallel computation we ensured from the beginning, along with the sampling, that the increment in the volume of the input dataset would not impact drastically the running time and convergence of the algorithm in a finite time. Analyzing the data with the SON allows to split the sampled dataset in multiple partitions, potentially without an upper bound, if the data were stored on a DFS on multiple clusters in the cloud. The workload will be so always split equally between each available node.

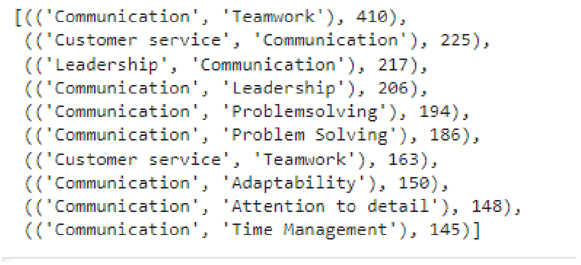
4 Description of the experiment

As it was anticipated before, the experiment has been structured showing firstly the two passes with the goal to extract the frequent singleton and the frequent couples over the partitions. Let's see the result below in the picture.



```
[('Communication', 1342),
 ('Teamwork', 839),
 ('Leadership', 640),
 ('Customer service', 582),
 ('Communication skills', 439),
 ('Customer Service', 396),
 ('Problem Solving', 383),
 ('Attention to detail', 339),
 ('Problemsolving', 332),
 ('Communication Skills', 324)]
```

Figure 2: Frequent singletons

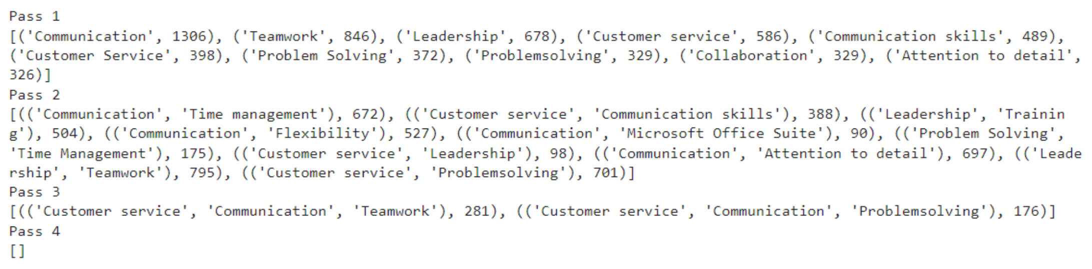


```
[(('Communication', 'Teamwork'), 410),
 (('Customer service', 'Communication'), 225),
 (('Leadership', 'Communication'), 217),
 (('Communication', 'Leadership'), 206),
 (('Communication', 'Problemsolving'), 194),
 (('Communication', 'Problem Solving'), 186),
 (('Customer service', 'Teamwork'), 163),
 (('Communication', 'Adaptability'), 150),
 (('Communication', 'Attention to detail'), 148),
 (('Communication', 'Time Management'), 145)]
```

Figure 3: Frequent couples

After computing the frequent couples the next step has been the iteration, until convergence, of all the possible size itemsets which every of their subsets are cointained in the frequent itemset extracted in the previous step. This is possible because we are sure that if a couple is frequent in a partition means that all of its element (singleton) are frequent as well. With this system we can easily sum the support for the itemsets without having to combine every element every time.

At the end the entire algorithm run in order to find the biggest sized frequent itemsets, in this case 3-triplets since no itemsets composed by four items has been evaluated as frequent considering the support threshold. Below it has shown the frequent itemsets for each iteration of the SON algorithm.

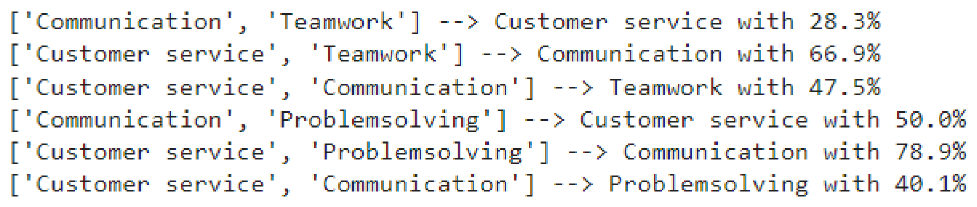


```
Pass 1
[('Communication', 1306), ('Teamwork', 846), ('Leadership', 678), ('Customer service', 586), ('Communication skills', 489),
('Customer Service', 398), ('Problem Solving', 372), ('Problemsolving', 329), ('Collaboration', 329), ('Attention to detail',
326)]
Pass 2
[(('Communication', 'Time management'), 672), (('Customer service', 'Communication skills'), 388), (('Leadership', 'Trainin
g'), 504), (('Communication', 'Flexibility'), 527), (('Communication', 'Microsoft Office Suite'), 90), (('Problem Solving',
'Time Management'), 175), (('Customer service', 'Leadership'), 98), (('Communication', 'Attention to detail'), 697), (('Leade
rship', 'Teamwork'), 795), (('Customer service', 'Problemsolving'), 701)]
Pass 3
[(('Customer service', 'Communication', 'Teamwork'), 281), (('Customer service', 'Communication', 'Problemsolving'), 176)]
Pass 4
[]
```

Figure 4: The top 10 for each frequent *i-itemset*

5 Results

The final objective of this analysis was to find the *association rules* between the different elements inside the baskets, in this case composed with job skills requested in the job offers. To compute the association rules it's been considered the last pass in the algorithm, so the frequent triplets found at the end of the execution, and then re-calculated the real support for each of them and the related 2-items subsets dividing again the computations between all the partitions in order to get the real sum of the support for each candidate itemset. This step has been done only for the first five frequent itemsets even though in our case only three triplets have been found frequents in the sample dataset. Here below all the association rules found have been shown.



```
['Communication', 'Teamwork'] --> Customer service with 28.3%
['Customer service', 'Teamwork'] --> Communication with 66.9%
['Customer service', 'Communication'] --> Teamwork with 47.5%
['Communication', 'Problemsolving'] --> Customer service with 50.0%
['Customer service', 'Problemsolving'] --> Communication with 78.9%
['Customer service', 'Communication'] --> Problemsolving with 40.1%
```

Figure 5: The association rules found for the frequent triplets

The results show that three of the most relevant job skills nowadays are customer service, teamwork and problem solving and that it's likely that also a good communication is requested to get the job.