Università degli Studi di Milano

Data Science for Economics

Market Basket Analysis
Algorithms for massive data project

Student:
Jing Wang

ACADEMIC YEAR 2023-2024

Contents

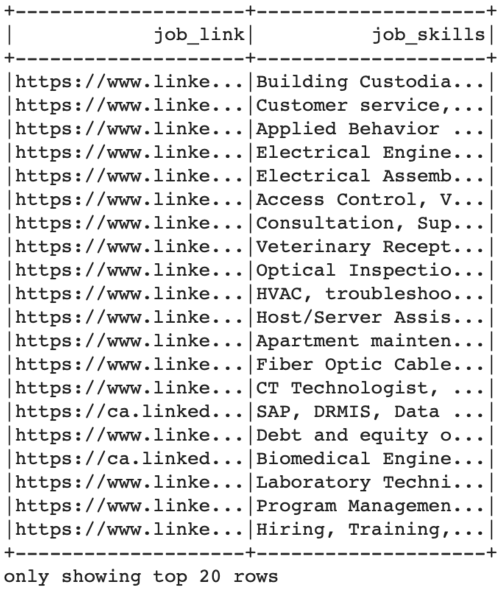
1 Introduction

In this project we are asked to implement a system finding frequent itemsets (aka market-basket analysis), analyzing the LinkedIn Jobs & Skills dataset which can be downloadable from the following link on kaggle: Dataset link.
The objective of this project is to implement a detector to retrieve the sets of jobs skills which are more frequent in some LinkedIn pages.
The dataset contains 1.3 million job listings scraped from LinkedIn in the year 2024. It is made up of three .csv files: **job_skills.csv**, **job_summary.csv**, **linkedin_job_postings.csv**. For this project, only the first file is considered.

2 About the dataset

The dataset is downloaded using my Kaggle API username and key.
As we can see from the following image, the dataset contained in job_skills.csv is composed of two columns:

- job_link is a column of strings relating to the LinkedIn job announcements.
- job_skills is a column of strings relating to the desirable skills attached to each job announcement.



```
+--------------------+--------------------+
|            job_link|          job_skills|
+--------------------+--------------------+
|https://www.linke...|Building Custodia...|
|https://www.linke...|Customer service,...|
|https://www.linke...|Applied Behavior ...|
|https://www.linke...|Electrical Engine...|
|https://www.linke...|Electrical Assemb...|
|https://www.linke...|Access Control, V...|
|https://www.linke...|Consultation, Sup...|
|https://www.linke...|Veterinary Recept...|
|https://www.linke...|Optical Inspectio...|
|https://www.linke...|HVAC, troubleshoo...|
|https://www.linke...|Host/Server Assis...|
|https://www.linke...|Apartment mainten...|
|https://www.linke...|Fiber Optic Cable...|
|https://www.linke...|CT Technologist, ...|
|https://ca.linked...|SAP, DRMIS, Data ...|
|https://www.linke...|Debt and equity o...|
|https://ca.linked...|Biomedical Engine...|
|https://www.linke...|Laboratory Techni...|
|https://www.linke...|Program Managemen...|
|https://www.linke...|Hiring, Training,...|
+--------------------+--------------------+
only showing top 20 rows
```

Figure 1: Dataset

First, I checked for the presence of null values in the dataset. Additionally, I focused exclusively on the second column for the analysis, as the first column contained irrelevant information for the analysis. The dataset retained **1,294,374** rows. Notably, one job listing requires 463 skills, while the average job listing on LinkedIn expects a candidate to have 21

skills to be considered competitive.
Due to the large size of the dataset and technological constraints in Google Colab, I sampled 1% of the data, which corresponds to **12,851** rows. This random subset was selected to make the analysis more manageable.

3 Data preparation

Before starting the main analysis, the **job_skills** column from the DataFrame was converted into a Resilient Distributed Dataset (rdd) for processing with PySpark, which is the Python version of the Apache Spark engine for dealing with Big Data. The rdd was then partitioned into six clusters. Using the map function, I transformed the rdd into rdd_new, which consists of strings of skills. The variable 'baskets' is created to convert the string of skills into a list of individual skill strings. The 'baskets' rdd is used then to create another rdd called RDD_F. In RDD_F, each skill is considered a single element, I discovered that there are 99,082 unique skills in the dataset analyzing this rdd. Each skill is then mapped to an index to construct the hash table, which is important for handling the data in a more efficient way and reducing the time complexity in retrieving the data during computations. This hash table is transformed into a Python dictionary, where the keys are skills and the values are their corresponding indexes. Finally, the lists in the rdd are converted into sets, providing the correct representation of the baskets.

4 The A-Priori algorithm

The A-Priori algorithm is commonly used in identifying frequent itemsets within large datasets.
This algorithm uses a minimum support threshold, that defines how frequent an itemset needs to be considered "interesting" for the analysis. Items or itemsets that appear less frequently than the threshold are discarded. Therefore, it is crucial to select a support that filters most of the data (to maintain the algorithm light) while not discarding interesting connections. In this analysis, i set it to 300, which means that an itemset is frequent if it appears more than 300 times. At first, I manually compute the passes required to identify different frequent itemsets using the A-Priori algorithm. Here's how each pass is executed:

First Pass:

- Map Transformation: I begin by performing a map transformation on the rdd called 'baskets'. For each item in each basket, I create a key-value pair where the key is the item_index (skill) and the value is 1.
- Reduce Operation: Next, I apply a reduce operation to aggregate these key-value pairs. Specifically, I sum all the 1s associated with each key (skill) to get the total count of occurrences for each skill across all baskets.
  As the result of the two operations, I obtain key-value pairs where the key is the skill index and the value is the count of that skill in the dataset.
- Filter all the items: At the end of the first pass, all items are filtered in order to keep only the those which have value greater than the support threshold.

61 frequent singletons have been identified and the result obtained is the following:

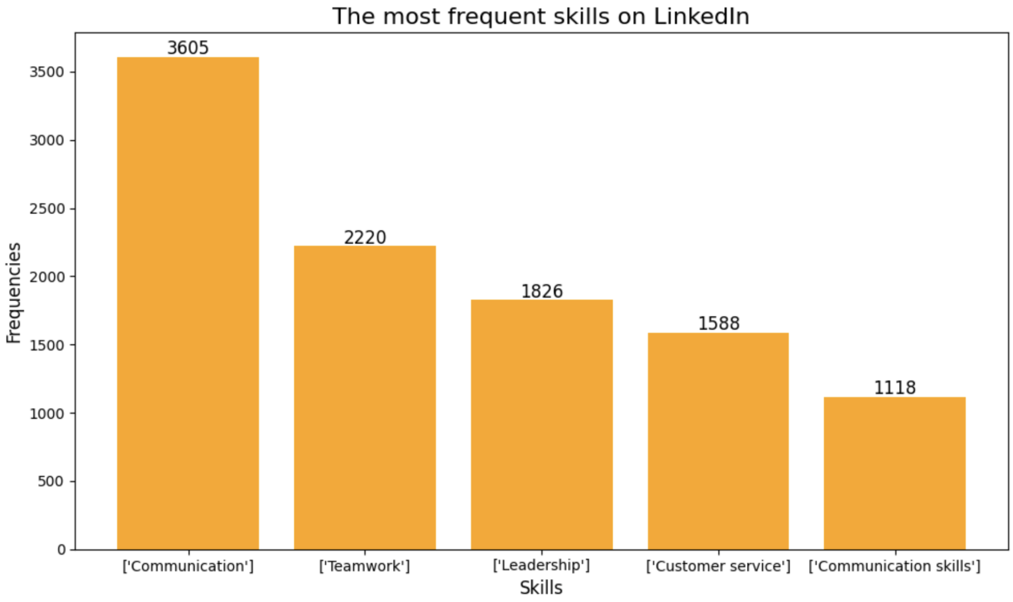


Figure 2: First pass

As we can see from the plot, the top 5 frequent skills retrieved are

- **Communication** with 3605 occurrences.
- **Teamwork** with 2220 occurrences.
- **Leadership** with 1826 occurrences.
- **Customer service** with 1588 occurrences.
- **Communication skills** with 1118 occurrences.

The first and the last ones are the same, but they are considered different because they are different strings.

Second pass:
the second pass is performed taking into account the **monotonicity** assumption which states that if a given itemset is frequent, all of its subsets must also be frequent.
Using MapReduce functions, I identify candidate pairs and their frequencies across all baskets, based on the singletons retrieved in the first pass. I then filter these pairs to retain only those that meet the support threshold. As a result, I obtained 33 pairs with more than 300 occurrences each.
In the following plot, there are 5 pairs of skills which are considered most relevant in the analysis.

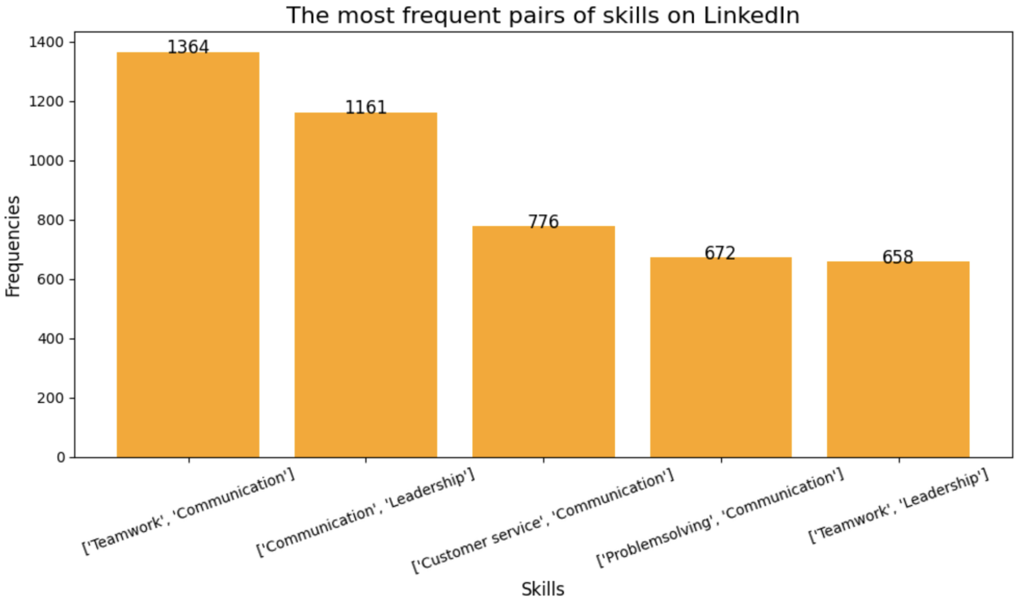


Figure 3: Second pass

The top 5 frequent skills pairs required by job announcements are

- **Teamwork and Communication** with 1364 occurrences.
- **Communication and Leadership** with 1161 occurences.
- **Customer service and Communication** with 776 occurences.
- **Problemsolving and Communication** with 672 occurences.
- **Teamwork and Leadership** with 658 occurences.

Third pass:
In the third pass, the goal is to generate and evaluate 3-item combinations (triplets) from the frequent 2-itemsets obtained in the second pass. Like in the second pass, this process leverages the monotonicity assumption, which ensures that if a 3-itemset is frequent, all of its 2-item subsets must also be frequent. This helps to reduce the candidate set significantly. The triple skills filtered by the support threshold are 4, which are shown as follows:

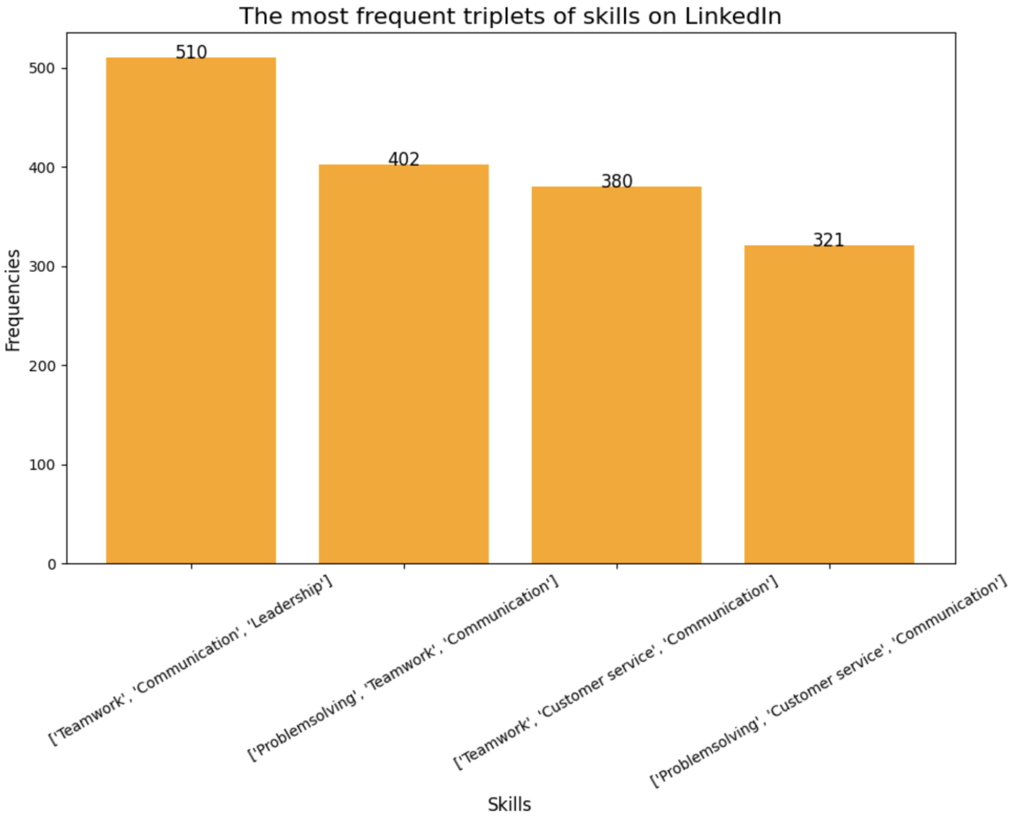


Figure 4: Third pass

The top 4 frequent skills triplets required by job offers are:

- **Teamwork, Communication and Leadership** with 510 occurrences.
- **Problemsolving, Teamwork and Communication** with 402 occurences.
- **Teamwork, Custom service and Communication** with 380 occurences.
- **Problemsolving,Custom service and Communication** with 321 occurences.

At the end of this analysis, I also defined an **.apriori()** function that automates the process of identifying the most frequent items and itemsets. This function follows the same steps as the manual process described earlier, but it simplifies the workflow by automatically generating frequent itemsets at each pass (singletons, 2-itemsets combinations, 3-itemsets combinations, etc.), discarding non-frequent candidates, and returning the most frequent itemsets. This way, the **.apriori()** function streamlines the entire A-Priori algorithm, making it more efficient. Even though I did not perform the fourth pass in the previous step, the results from this algorithm indicate that there are no 4-item combinations with frequencies exceeding the support threshold of 300.

5 Conclusions

All the results obtained in this analysis are based on the sampling process conducted at the beginning.
We can derive valuable insights from this study in the job market context: to remain competitive on LinkedIn, it is essential to possess skills that are closely related to **Communication, Teamwork, and Leadership**. These skills often appear together in frequent itemsets, indicating that they are highly evaluated by employers. Developing these interconnected skills can significantly enhance your professional profile, making you more attractive to potential employers and better equipped for leadership roles in team-driven environments.

6 References

Jure Leskovec, Anand Rajaraman, Jeff Ullman, ”Mining of Massive Datasets”

I declare that this material, which I now submit for assessment, is entirely my 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 work, and including any code produced using generative AI systems. I 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 or any other person for assessment on this or any other course of study.