Market Basket Analysis for the 1.3M LinkedIn Jobs & Skills

Hossein Akrami

Introduction

On this project I aim to analyze 1.3 million jobs that were advertised on LinkedIn at different times. This dataset is available through this link:

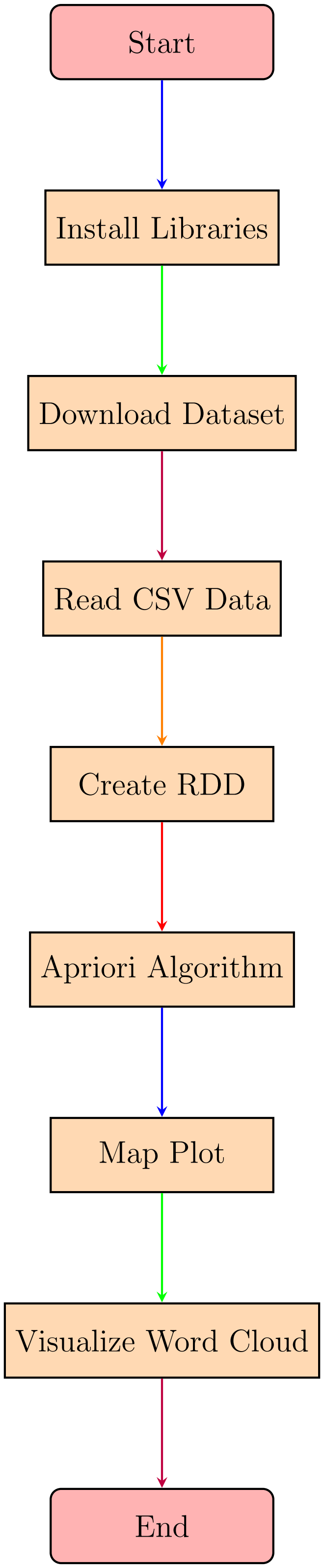
`https://www.kaggle.com/datasets/asaniczka/1-3m-linkedin-jobs-and-skills-2024`

In the dataset, there is one column in a specific CSV file named `job_skills.csv` in a compressed file. The aim is to find which skills usually appear together. For example, Google Cloud and Google BigQuery often appear together in LinkedIn job descriptions. Similarly, VBA and Microsoft Excel are commonly associated. However, we lack information about other fields such as mechanical engineering, energy engineering, or even vastly different fields like pottery, pastry, international relations, etc. We are interested in building a model that can identify these associations and provide further insights.

In this document, we will follow this pipeline:

1. Installing necessary libraries
2. Downloading the dataset from Kaggle
3. Reading the CSV Data
4. Creating an RDD (Resilient Distributed Dataset)
5. Applying the Apriori Algorithm
6. Mapping a plot
7. Visualizing a word cloud

Pipeline Flowchart



Environment and Setup

In this project, we have used Google Colab as the environment and Python 3 as the programming language.

The most necessary library used is:

```
!pip install –q pyspark==3.2.0
```

Then, we create a Spark session:

```
from pyspark.sql import SparkSession

spark = SparkSession.builder \
    .appName(”MarketBasketAnalysis”) \
    .config(”spark.driver.memory”, ”100g”) \
    .config(”spark.executor.memory”, ”12g”) \
    .getOrCreate()
```

After that, we download the dataset, which has a large size:

```
import os
os.environ['KAGGLE_USERNAME'] = ”xxxxxxx”
os.environ['KAGGLE_KEY'] = ”xxxxxxx”
!kaggle datasets download –d asaniczka/1–3m–linkedin–jobs–and–skills–2024
```

Then, we read the specific CSV file into a Spark DataFrame:

```
df = spark.read.option(”header”, ”true”).csv(”job_skills.csv”)
```

In this approach, we have read the data into a Spark DataFrame. This strategy takes a few minutes. The better strategy is to read the data into a normal Pandas DataFrame, because after unzipping, the data size is less than the limitation of the memory. Even, there is no need to unzip the file when working with Pandas because we can use the `zipfile` library for reading a CSV file into a Pandas DataFrame even without unzipping the file.

However, I have chosen this approach for **scalability**. In a case that we have much larger data and the data is not in-memory, I continue with the Spark DataFrame.

Removing Noise and Cleaning the Data

In the English language, there are many frequently appearing words like "as", "the", "of", "on", "in", "out", etc. The aim of this analysis is to detect pairs of words that usually appear together. However, these frequent words introduce a lot of noise into our analysis. Therefore, we need to remove them from the DataFrame.

To do this, we first need to download the list of stopwords using the Natural Language Toolkit (NLTK) library:

```
import nltk
nltk.download('stopwords')
```

The above line of code downloads a list of common English stopwords that we can remove from our text data. Stopwords are words that appear frequently in the language but do not contribute significantly to the meaning of the text, such as "and", "the", "is", etc.

Next, we create an RDD (Resilient Distributed Dataset) in PySpark. Again, it should be noted that we use this approach for the sake of **scalability**. One of the advantages of the RDD is that it can save data across different nodes, reducing the risk of data loss in the future. This approach ensures that even if a node fails, the data can be recovered from other nodes.

Introducing RDD (Resilient Distributed Dataset)

RDD, which stands for Resilient Distributed Dataset, is a core concept in PySpark. It is a special type of data structure that allows us to work with large datasets across multiple computers in a distributed environment. The key feature of RDD is that it is ”resilient”, meaning that if some parts of the data are lost due to a failure, they can be recovered. RDDs are designed to handle large amounts of data, making them suitable for big data processing.

In this project, we create an RDD to store and process our cleaned job skills data. Here is the line of code that creates an RDD from the cleaned skills column in our DataFrame:

```
skills_rdd = df2.select(”cleaned_skills”).rdd.flatMap(list)
```

This code is part of the PySpark library and converts the selected column of our DataFrame into an RDD, allowing us to perform distributed processing on the data.

Introducing the Apriori Algorithm

The Apriori algorithm is a popular method used in data mining to find frequent itemsets in a dataset. An itemset is a group of items that often appear together. For example, in a supermarket, ”bread” and ”butter” might be a frequent itemset because they are often bought together.

The Apriori algorithm helps us discover these associations by analyzing the frequency of different itemsets. This is useful in market basket analysis, where we want to understand which items or skills tend to appear together.

In this project, we use the Apriori algorithm to find patterns in the job skills data. Instead of writing the algorithm from scratch, we use the ‘mlxtend‘ library, which provides a ready-to-use implementation:

```
from mlxtend.frequent_patterns import apriori
```

This line of code imports the Apriori algorithm from the ‘mlxtend‘ library, making it easy for us to apply it to our dataset.

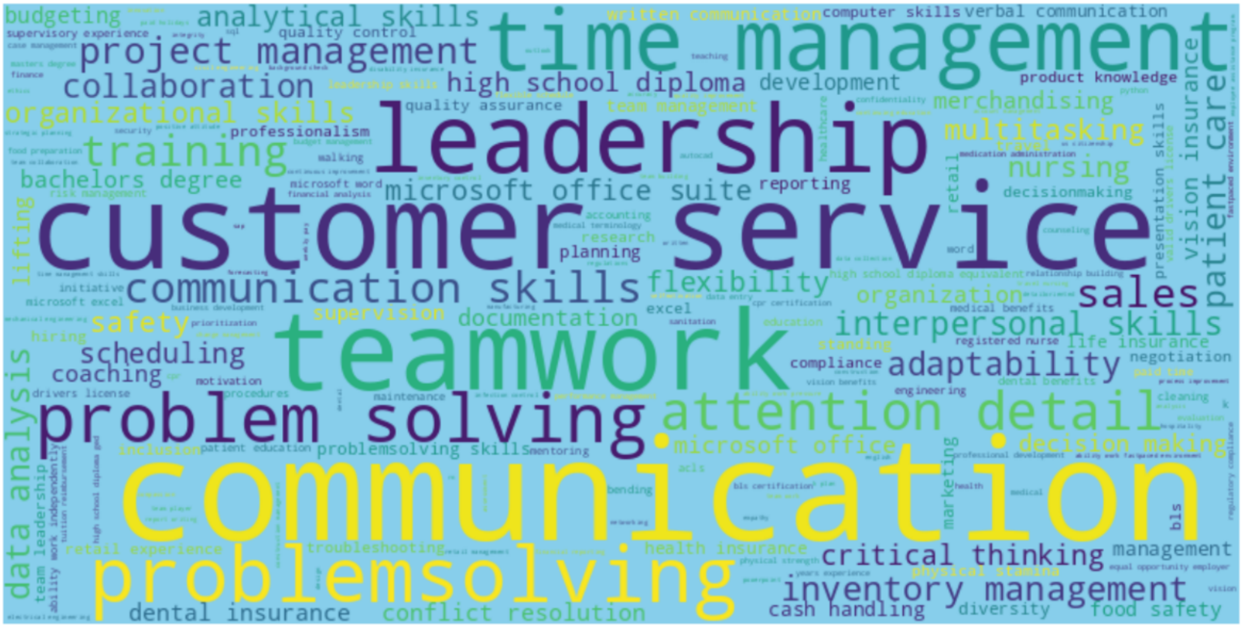
Word Cloud Visualization

Word clouds are a visual representation of text data, where the size of each word indicates its frequency or importance. In our project, we use a word cloud to visualize the most common skills found in the job listings.

We can generate a word cloud in Python using the ‘WordCloud‘ library:

```
from wordcloud import WordCloud
```

The word cloud image generated by our analysis will be included below:



figureWord cloud showing the most frequently-joined skills in LinkedIn job listings. The size of each word indicates its frequency or importance in the dataset.