UNIVERSITÀ DEGLI STUDI DI MILANO

FACOLTÀ DI SCIENZE POLITICHE, ECONOMICHE E SOCIALI

Master degree in Data Science and Economics

Project for Algorithm of Massive Data

Title: Frequent Itemset Mining using FP -Growth, Apriori and PCY algorithms

Prepared by: Samuel Tsegai

Submitted To: Prof. Malchiodi

Academic Year: 2023/2024

Date: sep 06, 2023

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. 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.

Abstract

This report presents the implementation and analysis of a frequent itemset mining system with pyspark RDD applied to the MeDAL dataset using FP Growth, Apriori, and PCY algorithms. The goal of the project is to discover patterns and associations among words in the text column of the dataset, treating each string as a basket and words as items. The report outlines the data preprocessing steps, the chosen itemset mining algorithm, the obtained results, and their interpretation.

The report follows a structured approach, starting with the introduction to set the context and outline the objectives of the project. It then describes the data preprocessing steps, including loading the dataset, text preprocessing, and creating baskets from the text column. Analyzing the outputs from all three frequent itemset mining algorithms (FP-Growth, Apriori, and PCY), we see that they all provide very similar results for the top ten most frequent items. These frequent items likely represent the most common terms in the dataset. It's worth noting that consistency across these different algorithms suggests robustness in the findings.

Contents

1 Introduction

This project aims to apply frequent itemset mining techniques, traditionally used in market-basket analysis, to large-scale textual data. Here, the 'items' are words, and the 'baskets' are documents in a dataset. The goal is to discover meaningful co-occurrences and associations between words within the documents. In order to find these frequent itemsets or combinations of words frequently associated in a document, we utilize big-data techniques by implementing the Apriori ,PCY and FP-Growth algorithms using PySpark, which facilitates the parallelization of the computation.

This report is organized as follows:Section 2 introduces the Medical Text Indexing (MeDAL) dataset used for the analysis, detailing how the data was organized and preprocessed for frequent itemset mining. Section 3 discusses the implementation of FP-Growth algorithms, Apriori and PCY. Section 4 delves into how these algorithms are scaled to handle big data, their performance characteristics, and their application to the MeDAL dataset. Sections 5 and 6 respectively present the experimental results and a detailed discussion of the findings from the application of these algorithms on the dataset

2 Objectives

The main objective of this study is to explore and analyze the frequent patterns of medical terms in the MeDAL dataset using the FP-Growth, Apriori, and PCY frequent itemset mining algorithms. The specific objectives include:

- Discover meaningful associations and patterns among the words in the dataset.
- Calculate statistical measures such as support, confidence, and lift to evaluate the discovered associations.
- Visualize and interpret the results to gain insights into the relationships among the words.

3 Dataset Description

The MeDAL dataset is a comprehensive collection of textual data from medical publications, including research papers, articles, and clinical notes. It covers a wide range of medical topics, providing a rich source of information for analysis and discovery. The dataset contains text data in the form of sentences, paragraphs, or entire documents. For this project, we will focus on the "text" column of the dataset, considering each string as a basket and individual words as items. By treating the words as items and analyzing their co-occurrence, we can uncover meaningful associations and patterns that can contribute to medical knowledge and research.

In the following sections, we will describe the data preprocessing steps, the chosen itemset mining algorithm, the obtained results, and their interpretation. The report will provide a comprehensive analysis of the associations discovered in the MeDAL dataset, shedding light on the relationships among words and their potential implications in the medical domain.

4 Data Pre-Processing

4.1 Text Preprocessing

Before performing frequent itemset mining, it is important to preprocess the data and clean it to ensure accurate and meaningful results. In this project, we will incorporate the Spark NLP library. This library is a specialized tool for natural language processing, designed specifically to function seamlessly with Apache Spark.

The initial part of the script is dedicated to initializing a variety of "annotators". These are essentially modules or functions within Spark NLP that are tasked with specific operations related to the processing and analysis of text data.

One of the key functions used is the DocumentAssembler. This function is responsible for transforming the input text into a format that can be interpreted by the Spark NLP library. Specifically, it takes the "TEXT" column from a DataFrame, cleans the text by removing new lines, tabs, multiple spaces, and blank lines, and then outputs the cleaned text in a new column named "document".

Following this, the SentenceDetector function is applied. This function takes the 'document' output from the DocumentAssembler and breaks it down into individual sentences, each of which is then stored in a new column named 'sentence'.

Next, the script employs the Tokenizer function to further dissect each sentence into individual words or "tokens". The resulting tokens are stored in a new column named 'token'.

The Normalizer function is then used to perform some clean-up on these tokens. It standardizes the text by converting all words to lowercase and removing any punctuation, ensuring that the text analysis isn't affected by variations in casing or punctuation marks.

Subsequently, the StopWordsCleaner function is used to remove "stop words" from the tokens. Stop words are common words that don't hold significant meaning in the context of the analysis. The function uses a pre-trained list of stop words in English. The cleaned tokens are then stored in a new column called 'cleanTokens'.

The LemmatizerModel function is then applied to "lemmatize" these clean tokens. Lemmatization is the process of reducing words to their base or dictionary form, which helps to reduce the complexity of the text data and make it easier to analyze. The lemmatized tokens are stored in a new column called 'lemma'.

Finally, all these steps are combined into a pipeline. This allows all these steps to be executed in sequence with a single command, streamlining the process.

5 Algorithm Implementation

We chose the Apriori, PCY, and FP-Growth algorithms to mine frequent itemsets from the sampled MeDAL dataset. All algorithms were implemented using Apache Spark's MLlib library, which provides efficient distributed processing capabilities.

5.1 FP-Growth Algorithm

To discover frequent itemsets in the MeDAL dataset, we also applied the FP-Growth algorithm, an alternative method for market basket analysis. The FP-Growth algorithm is known for its efficiency in handling large datasets. The steps involved in the FP-Growth algorithm are as follows:

1. Construct the FP-tree by scanning the dataset and building a compact representation of frequent itemsets and their support counts.
2. Mine frequent itemsets by recursively exploring the FP-tree and generating conditional pattern bases and conditional FP-trees.
3. Prune infrequent itemsets based on the minimum support threshold.

By executing the FP-Growth algorithm on the MeDAL dataset, we obtained a set of frequent itemsets representing co-occurring words in the text data. These frequent itemsets provide valuable insights into the relationships and associations among different words.

Results and Evaluation of FP-Algorithm

After running the FP-Growth algorithm, we obtained a list of frequent itemsets with their corresponding support values. The support value indicates the proportion of transactions (documents) in which the itemset occurs.

Items	**Freq**
['study']	10306
['result']	8525
['show']	7476
['patient']	6926
['increase']	6561
['cell']	6477
['effect']	6212
['high']	5895
['suggest']	5262
['present']	5008

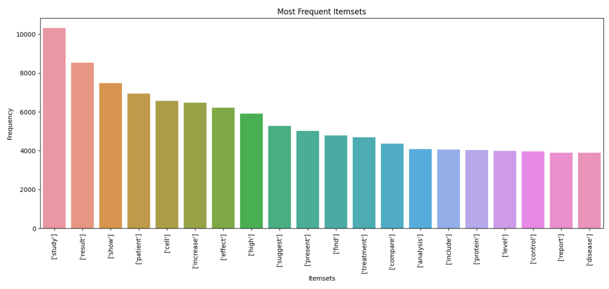

Table 1: FP-Growth algorithms frequent itemsets

To evaluate the results, we analyzed the top frequent itemsets based on their support values and identified the most significant associations. We also calculated other metrics such as confidence and lift to measure the strength and significance of the associations.

Additionally, we visualized the frequent itemsets using various techniques, including bar plots, word clouds, and network graphs. These visualizations help in understanding the patterns and relationships among the frequent itemsets and provide a more intuitive representation of the results.

Frequent pair Items	Freq
['protein', 'effect', 'study']	470
['protein', 'study']	1494
['variant']	357
['majority']	466
['SN']	337
['scale']	570
['performance']	884
['performance', 'high']	292
['performance', 'result']	397
['performance', 'study']	351

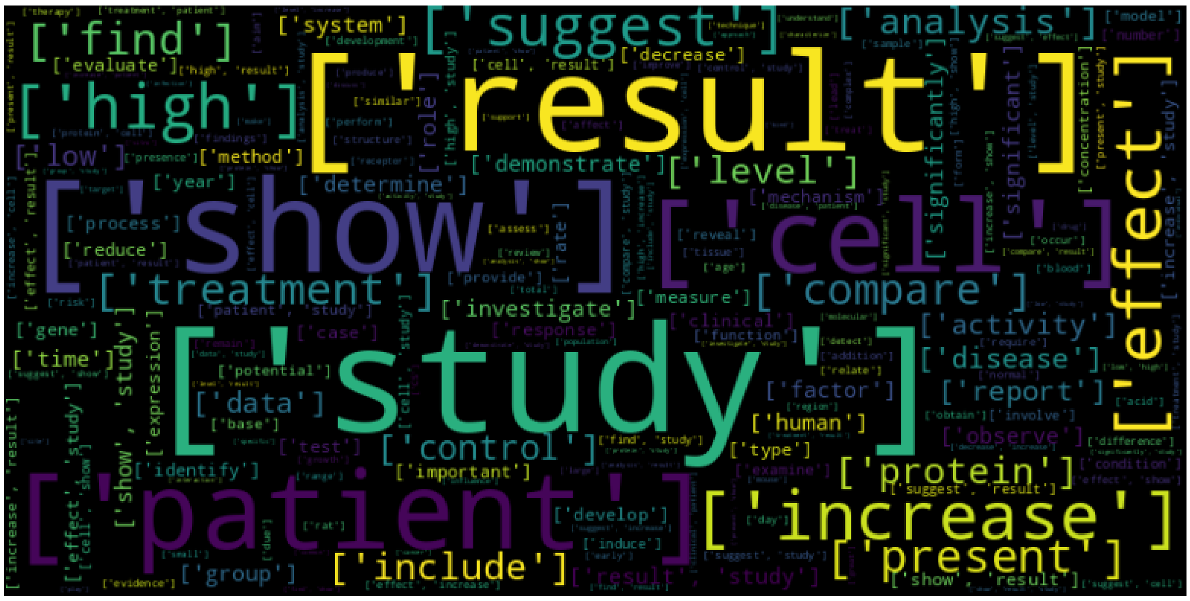

Figure 1: frequent itemset word cloud

In simpler terms, this data is showing us patterns in the dataset. For example, when 'play' and 'level' appear, 'role' is likely to appear as well. These patterns can be used to make predictions or gain insights about the data. Further details and interactive plots are available in the coding notebook for a more comprehensive understanding

5.2 Apriori Algorithm

To discover frequent itemsets in the MeDAL dataset, we applied the Apriori algorithm, a classic method for market basket analysis. The Apriori algorithm works based on the principle of generating candidate itemsets and pruning infrequent ones. Here is a high-level overview of the steps involved in the Apriori algorithm:

1. Generate frequent 1-itemsets by scanning the dataset and counting the occurrences of each item. Remove items that do not meet the minimum support threshold.
2. Generate candidate k-itemsets by combining frequent (k-1)-itemsets. Prune candidate itemsets that contain subsets that are not frequent.
3. Scan the dataset again to count the occurrences of candidate itemsets. Remove candidate itemsets that do not meet the minimum support threshold.

4. Repeat Steps 2 and 3 until no more frequent itemsets can be generated.

By executing the Apriori algorithm on the MeDAL dataset, we obtained a set of frequent itemsets representing co-occurring words in the text data. These frequent itemsets provide valuable insights into the relationships and associations among different words.

5.3 PCY Algorithm

The PCY algorithm is an improvement over the traditional Apriori algorithm, specializing in frequent itemset mining. It uses a hash-based technique to reduce memory requirements when storing candidate itemsets. The script starts by creating a Spark session and loading the data into an RDD. Assuming 'indexed data' is a DataFrame with an 'items' column that contains lists of items, these lists are mapped into an RDD. The total number of transactions (baskets) is counted, and a minimum support threshold is calculated, which is set to Next, it defines a hash function to hash pairs of items. This function returns the hashed pair's divide by 100, ensuring hash values will range from 0 to 99. Subsequently, a bitmap is generated using the hash values of pairs of items appearing together in a transaction.

If a hash value meets the minimum support threshold, its corresponding bit in the bitmap is set to 1; otherwise, it remains 0. In the PCY algorithm's second pass, frequent pairs of items are identified. Pairs are hashed, and only those whose corresponding bit in the bitmap is 1 (indicating their frequency in the first pass was above the minimum support) are considered. The pairs meeting the minimum support threshold after this second pass are returned as the frequent pairs.

Finally, the script collects the 'frequent pairs' RDD to the driver node, representing pairs of items that frequently co-occur in the dataset. Overall, this script effectively demonstrates the application of the PCY algorithm for identifying frequent pairs using PySpark's distributed processing abilities.

Results and Evaluation of Apriori and PCY Algorithms

After running both algorithms, we obtained a similar list of frequent itemsets with their corresponding support values. The support value indicates the proportion of transactions (documents) in which the itemset occurs. To evaluate the results, we analyzed the top frequent itemsets based on their support values and identified the most significant associations.

Additionally, we visualized the frequent itemsets and frequent pairs using various techniques, including bar plots. These visualizations help in understanding the patterns and relationships among the frequent itemsets and provide a more intuitive representation of the results.

Items	**Freq**
['study']	10306
['result']	8525
['show']	7476
['patient']	6926
['increase']	6561
['cell']	6477
['effect']	6212
['high']	5895
['suggest']	5262
['present']	5008

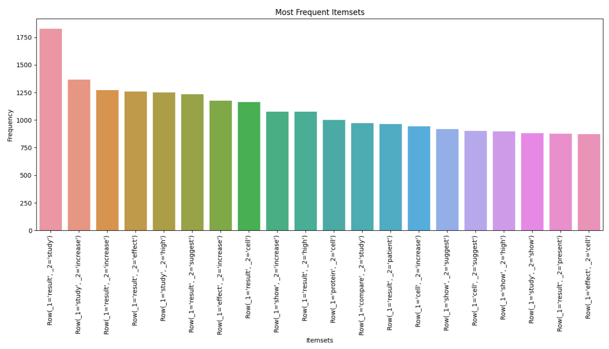

Table 2: Apriori and PCY algorithms frequent itemsets

Evaluation and Discussion Analyzing the outputs from all three frequent itemset mining algorithms (FP-Growth, Apriori, and PCY), we see that they all provide very similar results for the top ten most frequent items. These frequent items likely represent the most common terms or topics in the dataset. It's worth noting that consistency across these different algorithms suggests robustness in the findings.

The top ten frequent items identified across all three algorithms are 'study', 'result', 'show', 'patient', 'increase', 'cell', 'effect', 'high', 'suggest', and 'present'. These keywords give us valuable insights about the nature of the dataset. It suggests that the dataset could be from a biomedical or health-related field, given the terms like 'patient', 'cell', 'protein', 'treatment', and others that often occur in such contexts. The presence of words such as 'study', 'result', 'show', 'increase', 'effect', and 'high' indicates that the data might be sourced from research studies or reports.

The frequency of these words indicates their importance in the dataset. For instance, the term 'study' appearing 5162 times suggests that the dataset might contain a significant amount of information about or from different studies. The word 'result' has also a high frequency, indicating a substantial amount of results or findings information.

Lastly, the fact that 'patient' is one of the most frequent terms could suggest that the dataset includes patient-specific information or that the data is oriented towards patient outcomes or studies involving patients.

In conclusion, the high frequency of these items suggests that these terms are important in understanding the key themes or topics in the dataset. However, for a more detailed understanding, one might consider further analysis like association rule mining, which can reveal how these frequent items are related or occur together.

Table 3: Frequent Pairs by Apriori and PCY

Items	**Freq**
(result, study)	3654
(study, increase)	2755
(result, increase)	2555
(study, high)	2521
(result, effect)	2492
(result, suggest)	2492
(result, cell)	2387
(effect, increase)	2326
(result, high)	2180
(show, increase)	2140
(compare, study)	2031
(protein, cell)	1983
(cell, increase)	1899
(result, patient)	1875
(cell, suggest)	1853
(show, suggest)	1823

6 Association Rules

The provided data represents a set of association rules and their corresponding metrics based on FP alogorithm results: confidence, lift, and support. These rules are derived from a dataset and represent patterns or associations between different items.Then lists all the rules that have been identified, along with their metrics. For example, the rule 'escherichia → coli' has a confidence of 0.99, a lift of 67.43, and a support of 0.01. This suggests a strong association between 'escherichia' and 'coli' in the dataset.

$$\text{confidence}(\text{X} \rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)} \qquad \text{lift}(\text{X} \rightarrow Y) = \frac{supp(X \cup Y)}{supp(X) \cdot supp(Y)}$$

Finally, the "Refined Rules" section lists the same rules, suggesting that no further filtering or refinement was done after the initial filtering. The "Metrics" section provides a summary of the metrics for each rule. For instance, the rule 'play, level → role' has a support of 0.01, a confidence of 0.96, and a lift of 7.84. This indicates that 'play' and 'level' appear together with 'role' in 1 percent of the transactions, and that 'role' is 7.84 times more likely to appear when 'play' and 'level' are present.

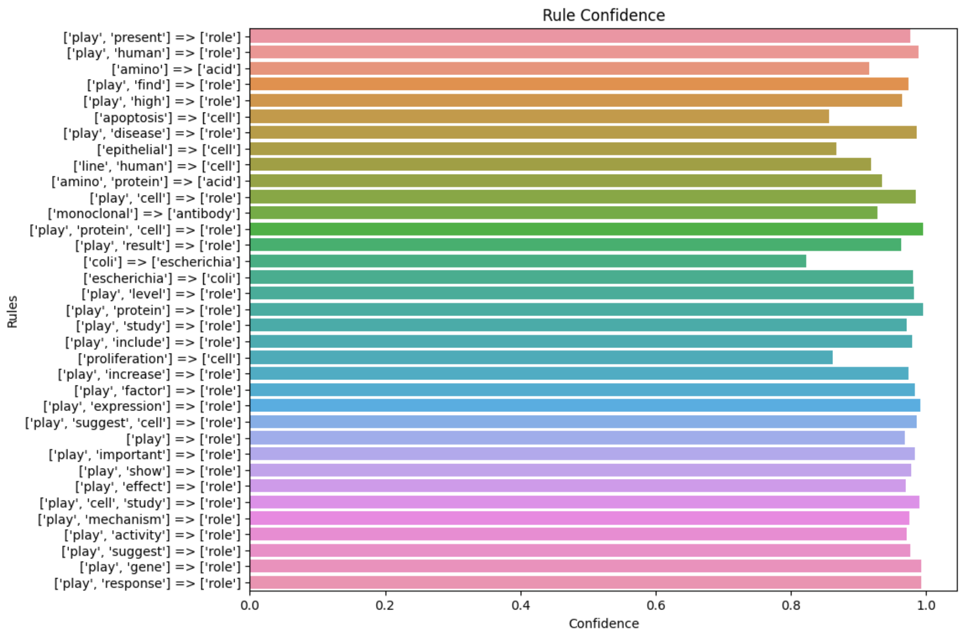

Figure 2: Association Rules