Algorithm for massive data

Market-Basket analysis on Medal Dataset

Marco Casagli 944665

October 2023

1 Introduction

The scope of the project I have realized consists in doing a Market basket Analysis on the MeDal Kaggle dataset using the text field as basket and the words contained in the text field as items.

The Market Basket Analysis (MBA) is an analysis to identify associations between items present in baskets, containers of items. It was first used in customer purchases especially in the supermarkets where it showed associations like people buying dyapers were buying even beer. This insight can be very useful for the managers who can make promotions in order to increase sales and to make target marketing strategies.

Market Basket Analysis can be used in different contexts like e-commerce (look at amazon suggesting products to be bought in combination), healthcare, telecommunications etc.

The project has been developed in Pyhton using the Colab platform and using Spark in order to speed up the execution time. Even though I have used Spark the execution time is computational intensive therefore I have used a sample of the dataset. Spark is an open source system by the Apache foundation designed to process large-scale amount of data. The key differentiator of Spark is the in memory computing and distributed computing

2 Data-set

The Kaggle Medal dataset stays for **Me**dical **D**ataset for **A**bbreviation Disambiguation for Natural **L**anguage Understanding (MeDAL) is a large medical text dataset curated for abbreviation disambiguation, designed for natural language understanding pre-training in the medical domain.

The entire dataset is made of 12.000.000 of records and the text field itself contains many words like 60 or more. Multiplying the number of words by the number of records to estimate the number of items we have 720.000.000 of items.

Therefore, is necessary to work on a subset. I've chosen to work on 150.000 records.

3 Logic of the Program

The logic of the program consists in first installing the necessary python libraries to download the dataset from Kaggle, preprocess the dataset to make it fit to the further analysis using FPGrowth and Apriori, install the Fpgrowth and Apriori libraries and run the mentioned algorithm.

The program uses Spark, installing it and opening a session and using SQL context to use Spark SQL.

4 Data Prepocessing

Once the dataset is downloaded from Kaggle and a subset of the Medal Dataset is loaded into a Spark Data frame.

In order to keep the execution time of the Fpgrowth Algorithm in an acceptable time I've limited the number of records to 150.000. Considered that the text field has typically 60 words the number of items to be processed by the algorithm is in the order of 9.000.000 of items.

Even applying this reduced version of the dataset, the FPgrowth algorithm takes 45 minutes to be executed.

In order to prepare the data to optimally execute the 2 algorithms (FPGrowth and Apriori,) covered in this project the dataset is subject to the following steps_

- The text field of the dataset is tokenized (divided in words)
- The words with a length less than 8 characters have been removed. This is a filter I have applied to exclude prepositions, connection words and words not meaningful for the analysis that is mainly for long Medical terms.
- Finally the dataset has been rendered unique removing duplicates

5 Frequent Pattern Algorithm

To identify the most frequent words or combination of words (items) in the baskets we apply the FpGrowth Algorithm that represents an efficient version of the Apriori algorithm.

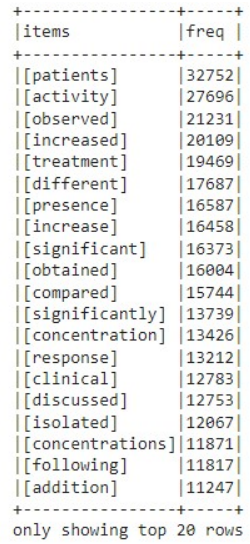
The steps used by the algorithm are here listed:

- Scan the dataset to identify the frequent items and calculate the frequency
- Construct the FP-tree with frequent associations
- Extracted the frequent item-sets from the FP-tree.

The algorithm is executed using a minimum support of 0.02 in order to generate the relative association rules

```
+--------------------+-----+
|items               |freq |
+--------------------+-----+
|[metabolic]         |3243 |
|[suggests]          |4875 |
|[patients]          |32752|
|[cultures]          |4828 |
|[activity]          |27696|
|[activity, patients]|3656 |
|[combined]          |3235 |
|[observed]          |21231|
|[observed, activity]|4948 |
|[observed, patients]|4309 |
|[appearance]        |3229 |
|[structure]         |4742 |
|[affected]          |4741 |
|[increased]         |20109|
|[increased, observed]|3942 |
|[increased, activity]|5556 |
|[increased, patients]|4340 |
|[affinity]          |3227 |
|[comparison]        |4675 |
|[treatment]         |19469|
+--------------------+-----+
only showing top 20 rows
```

Figure 1: Item frequencies



items	freq
[patients]	32752
[activity]	27696
[observed]	21231
[increased]	20109
[treatment]	19469
[different]	17687
[presence]	16587
[increase]	16458
[significant]	16373
[obtained]	16004
[compared]	15744
[significantly]	13739
[concentration]	13426
[response]	13212
[clinical]	12783
[discussed]	12753
[isolated]	12067
[concentrations]	11871
[following]	11817
[addition]	11247

only showing top 20 rows

Figure 2: Item sorted by frequencies

6 A- priori Algorithm

A-priori algoritmi s a two step approach.

In the first step it keeps just the items that are greater than the minimum support than leveraging the monotonicity property it prunes out subsets of candidates that do not meet the minimum support.

In the second step it creates association rules for the frequent items.

```
Frequent Itemset: frozenset({'patients'})
Support: 0.21877777777777777
Association Rules:
Rule: [] -> ['patients']
Confidence: 0.21877777777777777
Lift: 1.0
---
==================================================
Frequent Itemset: frozenset({'activity', 'addition'})
Support: 0.02011111111111111
Association Rules:
Rule: ['addition'] -> ['activity']
Confidence: 0.2679100059206631
Lift: 1.4371997694975074
---
==================================================
Frequent Itemset: frozenset({'compared', 'activity'})
Support: 0.021444444444444443
Association Rules:
Rule: ['compared'] -> ['activity']
Confidence: 0.2041032148900169
Lift: 1.0949090624129176
---
==================================================
Frequent Itemset: frozenset({'activity', 'concentration'})
Support: 0.023933333333333334
Association Rules:
Rule: ['concentration'] -> ['activity']
Confidence: 0.26767739530259727
Lift: 1.4359519328386334
---
==================================================
Frequent Itemset: frozenset({'activity', 'concentrations'})
Support: 0.021622222222222223
Association Rules:
Rule: ['concentrations'] -> ['activity']
Confidence: 0.27420036635197975
Lift: 1.470944326856898
```

Figure 3: Association Rules Apriori Algorithm

7 Conclusion

The MeDal dataset contains millions of records. In order to do a Market Basket Analysis on the text field it must be pruned, and the text filed must be subject to data preprocessing. Even following this approach, the execution time is quite intensive. I have applied two algorithms FP-Growth and A-Priori. The two algorithms are running on two different samples of the original Medal dataset, 150000 records for FP_Growth and 10.000 records for Apriori. The scope of the project was to demonstrate the possibility to use different algorithms to demonstrate their different approach in identifying frequent associations.

8 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.“